

PIC18F2455/2550/4455/4550 数据手册

采用纳瓦技术的 28/40/44 引脚高性能增强型闪存 USB 单片机



PIC18F2455/2550/4455/4550

采用纳瓦技术的28/40/44引脚高性能 增强型闪存 USB 单片机

通用串行总线特性：

- 兼容 USB V2.0
- 低速 （1.5 Mb/s）和全速 （12 Mb/s）
- 支持控制、中断、等时和批量数据传输
- 支持高达 32 个端点 （双向 16 对）
- USB 具有 1KB 双重存取 RAM
- 内置稳压器的片上 USB 收发器
- 外部 USB 收发器接口
- USB 并行传输 （仅 40/44 引脚器件）采用并行通信端口 （SPP）

功耗管理模式：

- 运行：CPU 工作，外设打开
- 空闲：CPU 不工作，外设打开
- 休眠：CPU 不工作，外设关闭
- 处于空闲模式时电流降至 5.8 µA （典型值）
- 处于休眠模式时电流降至 0.1 µA （典型值）
- Timer1 振荡器：1.1 µA （典型值）、 32 kHz 和 2V
- 看门狗定时器：2.1 µA （典型值）
- 振荡器双速启动

灵活的振荡器结构：

- 4 种晶振模式，包括用于 USB 的高精度 PLL
- 两种外部时钟模式，频率最高为 48 MHz
- 内部振荡器电路：
  - 8 种可由用户选择的频率，从 31 kHz 到 8 MHz
  - 用户可对该电路进行调节以补偿频率漂移
- 辅助振荡器使用 Timer1 （工作频率为 32kHz）
- 双振荡器选项，允许单片机和 USB 模块运行在不同的时钟速率下
- 故障保护时钟监视器
  - 在时钟停止时可使器件安全断电

外设特点：

- 高灌 / 拉电流 25 mA/25 mA
- 3 个外部中断
- 4 个定时器模块 （Timer0 到 Timer3）
- 2 个捕捉 / 比较 /PWM （CCP）模块：
  - 捕捉采用一个 16 位寄存器，最大分辨率 6.25 ns （T_{CY}/16）
  - 比较采用一个 16 位寄存器，最大分辨率 100 ns （T_{CY}）
  - PWM 输出：PWM 分辨率为 1 到 10 位
- 增强型捕捉 / 比较 /PWM （ECCP）模块：
  - 多种输出模式
  - 可选择的极性
  - 可编程的死区时间
  - 自动关闭和自动重启
- 增强型 USART 模块：
  - 支持 LIN 总线
- 主同步串口 （Master Synchronous Serial Port，MSSP）模块支持 3 线 SPI （总共 4 种模式）和 I^2C™ 主从模式
- 10 位最多 13 路通道模数转换器模块 （A/D），采集时间可编程
- 两个输入复用的模拟比较器

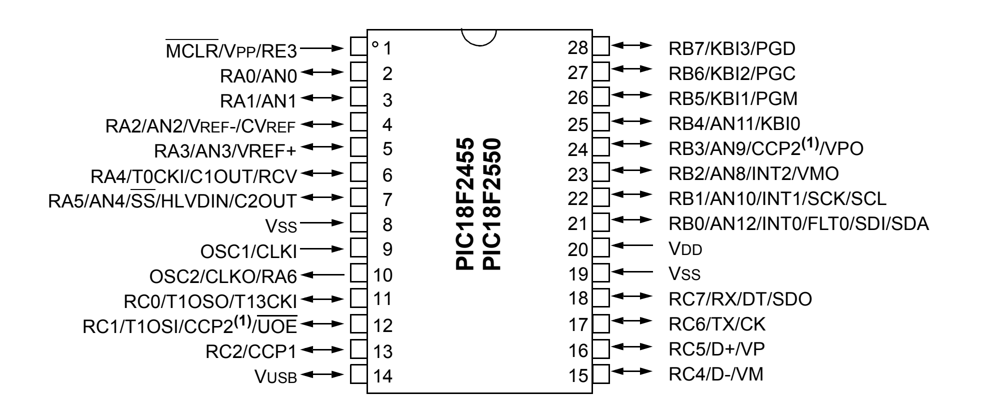
单片机的特殊性能：

- 优化的 C 语言编译器架构，带有可选的扩展指令集
- 增强型闪存程序存储器， 100,000 次擦写周期 （典型值）
- 数据 EEPROM 存储器， 1,000,000 次擦写周期 （典型值）
- 闪存 / 数据 EEPROM 数据保存时间：大于 40 年
- 可在软件控制下自编程
- 中断优先级
- 8 x 8 单周期硬件乘法器
- 扩展的看门狗定时器 （Watchdog Timer， WDT）：
  - 可编程周期从 41 ms 到 131s
- 可编程代码保护
- 通过两个引脚进行单 5 V 供电的在线串行编程 （In-Circuit Serial Programming™， ICSP™）
- 通过两个引脚进行在线调试 （In-Circuit Debug， ICD）
- 可选的专用 ICD/ICSP 端口 （仅 44 引脚器件）
- 宽工作电压范围：2.0V 到 5.5V

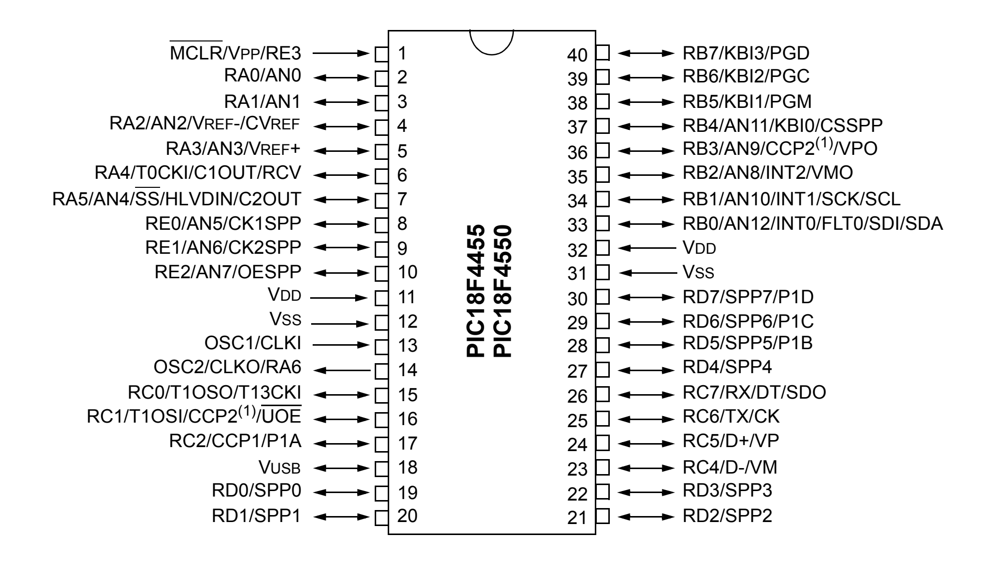
器件	程序存储器		数据存储器		I/O	10 位 A/D（通道数）	CCP/ECCP（PWM）	SPP	MSSP		EAUSART	比较器	8/16 位定时器
	闪存（字节）	单字指令数	SRAM（字节）	EEPROM（字节）					SPI	主控 I^2C™			
PIC18F2455	24K	12288	2048	256	24	10	2/0	无	有	有	1	2	1/3
PIC18F2550	32K	16384	2048	256	24	10	2/0	无	有	有	1	2	1/3
PIC18F4455	24K	12288	2048	256	35	13	1/1	有	有	有	1	2	1/3
PIC18F4550	32K	16384	2048	256	35	13	1/1	有	有	有	1	2	1/3

引脚图

28 引脚 PDIP，SOIC



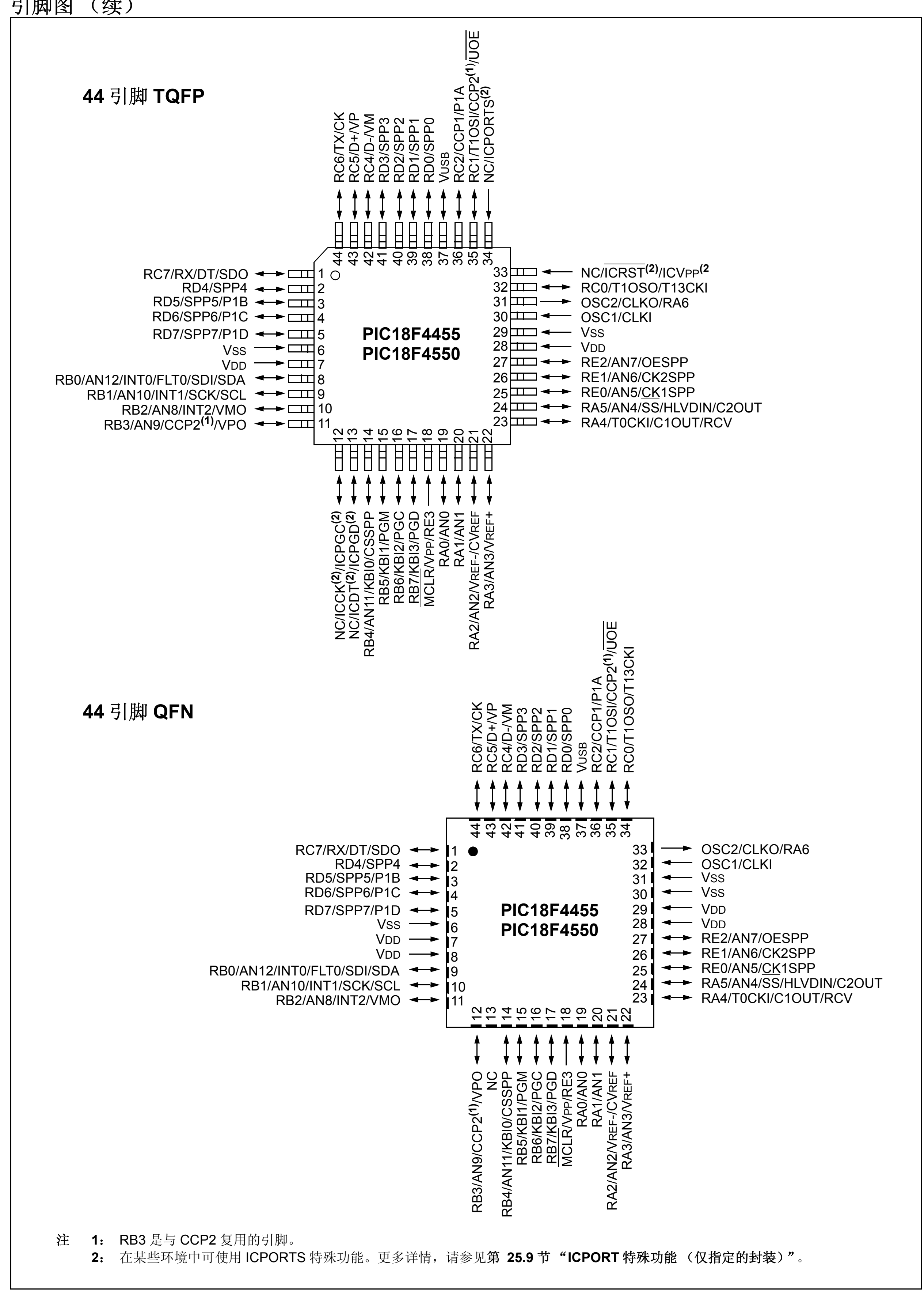
40 引脚 PDIP



注 1： RB3 是与 CCP2 复用的引脚。

 初稿

引脚图 （续）



注 1： RB3 是与 CCP2 复用的引脚。
2： 在某些环境中可使用 ICPORTS 特殊功能。更多详情，请参见**第 25.9 节 “ICPORT 特殊功能 （仅指定的封装）”**。

目录

初稿

致客户

我们旨在提供最佳文档供客户正确使用 Microchip 产品。为此，我们将不断改进出版物的内容和质量，使之更好地满足您的要求。出版物的质量将随新文档及更新版本的推出而得到提升。

如果您对本出版物有任何问题和建议，请通过电子邮件联系我公司 TRC 经理，电子邮件地址为 **CTRC@microchip.com**，或将本数据手册后附的 **《读者反馈表》**传真到 86-21-5407 5066。我们期待您的反馈。

最新数据手册

欲获得本数据手册的最新版本，请查询我公司的网站：

http://www.microchip.com

查看数据手册中任意一页下边角处的文献编号即可确定其版本。文献编号中数字串后的字母是版本号，例如：DS30000A是DS30000的 A 版本。

勘误表

现有器件可能带有一份勘误表，描述了实际运行与数据手册中记载内容之间存在的细微差异以及建议的变通方法。一旦我们了解到器件 / 文档存在某些差异时，就会发布勘误表。勘误表将注明其所适用的硅片版本和文件版本。

欲了解某一器件是否存在勘误表，请通过以下方式之一查询：

- Microchip 网站：http://www.microchip.com
- 当地 Microchip 销售办事处 （见最后一页）

在联络销售办事处时，请说明您所使用的器件型号、硅片版本和数据手册版本 （包括文献编号）。

客户通知系统

欲及时获知 Microchip 产品的最新信息，请到我公司网站 **www.microchip.com** 上注册。

注：

1.0 器件概述

该文档包含针对以下器件的信息：

- PIC18F2455
- PIC18F2550
- PIC18F4455
- PIC18F4550
- PIC18LF2455
- PIC18LF2550
- PIC18LF4455
- PIC18LF4550

该系列具备所有PIC18单片机固有的优点——即优惠的价格和出色的计算性能，还具有高耐久性和增强型闪存程序存储器。除此之外，PIC18F2455/2550/4455/4550 系列还引进了增强的功能，使得该系列单片机成为许多高性能和节能应用的明智选择。

1.1 新内核特性

1.1.1 纳瓦技术

PIC18F2455/2550/4455/4550 系列的所有器件具有一系列能在工作时显著降低功耗的功能。主要包括以下几项：

- **备用运行模式：** 通过将 Timer1 或内部振荡器电路作为单片机时钟源，可使代码执行时的功耗大约降低 90%。
- **多种空闲模式：** 单片机还可在其 CPU 内核禁止而外设仍然运行的情况下工作。处于这些状态时，功耗能降得更低，只有正常工作时的 4%。
- **动态模式切换：** 在器件工作期间可由用户代码调用功耗管理模式，允许用户将节能的理念融入到他们的应用软件设计中。
- **关键模块低功耗：** Timer1 和看门狗定时器模块的功耗需求可降到最低。请参见**第 28.0 节 "电气规范"**了解具体数值。

1.1.2 通用串行总线 （USB）

PIC18F2455/2550/4455/4550 系列器件内置了一个功能齐全的通用串行总线通讯模块，该模块符合 USB 规范 2.0 版。此模块支持所有允许的数据传输类型的低速和全速通信。它还具备了片内收发器和 3.3V 稳压器，也支持外部收发器和稳压器。

1.1.3 多个振荡器选项和特点

PIC18F2455/2550/4455/4550 系列的所有器件提供 12 种不同的振荡器选项，使用户在开发应用硬件时有很大的选择范围。这些选项包括：

- 4 种晶振模式，使用晶振或陶瓷谐振器。
- 4 种外部时钟模式，提供使用两个引脚 （振荡器输入引脚和四分频时钟输出引脚）或一个引脚 （振荡器输入引脚，四分频时钟输出引脚作为通用 I/O 引脚）的选项。
- 1 个内部振荡器模块，它提供 1 个 8 MHz 时钟 （精度± 2%）和 1 个 INTRC 时钟源 （振荡频率大约为 31 kHz，温度和 VDD 变化时频率保持稳定），以及 6 种用户可选择的时钟频率范围 （从 125 kHz 到 4 MHz），共有 8 种时钟频率可供选用。此选项可以空出一个振荡器引脚作为额外的通用 I/O 引脚。
- 一个锁相环 （Phase Lock Loop， PLL）倍频器，可用于高速晶振和外部振荡模式，使时钟速度达到 4 MHz 到 48 MHz。
- 异步双时钟运行模式，允许 USB 模块以高频振荡器作为时钟源运行，而单片机的其他部分由内部低功耗振荡器提供时钟。

除了可被用作时钟源，内部振荡器电路还提供了一个稳定的参考源，增加了以下功能以使器件更安全地工作：

- **故障保护时钟监视器：** 该选项不停地监视主时钟源，将其与内部振荡器提供的参考信号作比较。如果主时钟发生了故障，单片机会将时钟源切换为内部振荡器电路，使器件可继续低速工作或安全地关闭。
- **双速启动：** 该功能允许在上电复位或从休眠模式唤醒时将内部振荡器用作时钟源，直到主时钟源可正常工作为止。

1.2 其他特性

- **存储器耐擦写能力**：程序存储器和数据 EEPROM 的增强型闪存单元经测试，能经受数千次擦 / 写，程序存储器最高可达 100,000 次， EEPROM 最高可达 1,000,000 次。如果不刷新，数据保存期保守地估计在 40 年以上。
- **自编程性**：这些器件能在内嵌软件控制下对各自的程序存储空间进行写操作。通过使用位于受保护的引导模块 （程序存储器顶端）中的引导加载子程序，应用程序可实现现场自我更新。
- **扩展的指令集**： PIC18F2455/2550/4455/4550 系列在 PIC18 指令集的基础上进行了扩展，添加了 8 条新指令和变址寻址模式。此扩展可作为一个器件配置选项，它是为优化使用高级语言 （如 C 语言）开发的重入代码而特别设计的。
- **增强型 CCP 模块**：在 PWM 模式下，该模块提供 1、 2 或 4 路调制输出来控制半桥和全桥驱动器。其他功能包括自动断电 （能在中断或其他条件下禁止 PWM 输出）和自动重启 （能在禁止条件被清除时再次激活输出）。
- **增强型可寻址 USART**：此串行通信模块可进行标准的 RS-232 通信并支持 LIN 总线协议。其他增强的功能包括自动波特率检测和分辨率更高的 16 位波特率发生器。当单片机使用内部振荡器电路时，EUSART 为与外界通信的应用程序提供稳定的通信方式，而不需要使用外部晶振也无需额外的功耗。
- **10 位 A/D 转换器**：此模块具备可编程采集时间，从而不必在选择通道和启动转换之间等待一个采样周期，因而减少了代码开销。
- **专用 ICD/ICSP 端口**：这些器件包含一些不与其他单片机功能复用的编程引脚和调试引脚。选择特定的封装类型可以使用这些引脚。这一功能允许用户开发对 I/O 要求较高的应用，同时能够进行在线编程和调试。

1.3 系列中各产品的具体信息

PIC18F2455/2550/4455/4550 系列器件有 28 引脚和 40/44 引脚两种封装形式。图 1-1 和图 1-2 分别为这两类器件的框图。

这两类器件在以下 5 个方面存在差异：

1. 闪存程序存储器 （PIC18FX455 器件为 24 KB，PIC18FX550 器件为 32 KB）。
2. A/D 通道 （28 引脚器件有 10 路通道， 40/44 引脚器件有 13 路通道）。
3. I/O 端口（28 引脚器件上有 3 个双向端口和 1 个输入端口，40/44 引脚器件上有 5 个双向端口）。
4. CCP 和增强型 CCP （28 引脚器件有 2 个标准 CCP 模块， 40/44 引脚器件有 1 个标准 CCP 模块和 1 个 ECCP 模块）。
5. 并行通信端口 （仅 40/44 引脚器件上存在）。

该系列器件的其他功能都是相同的。表 1-1 汇总了这些功能。

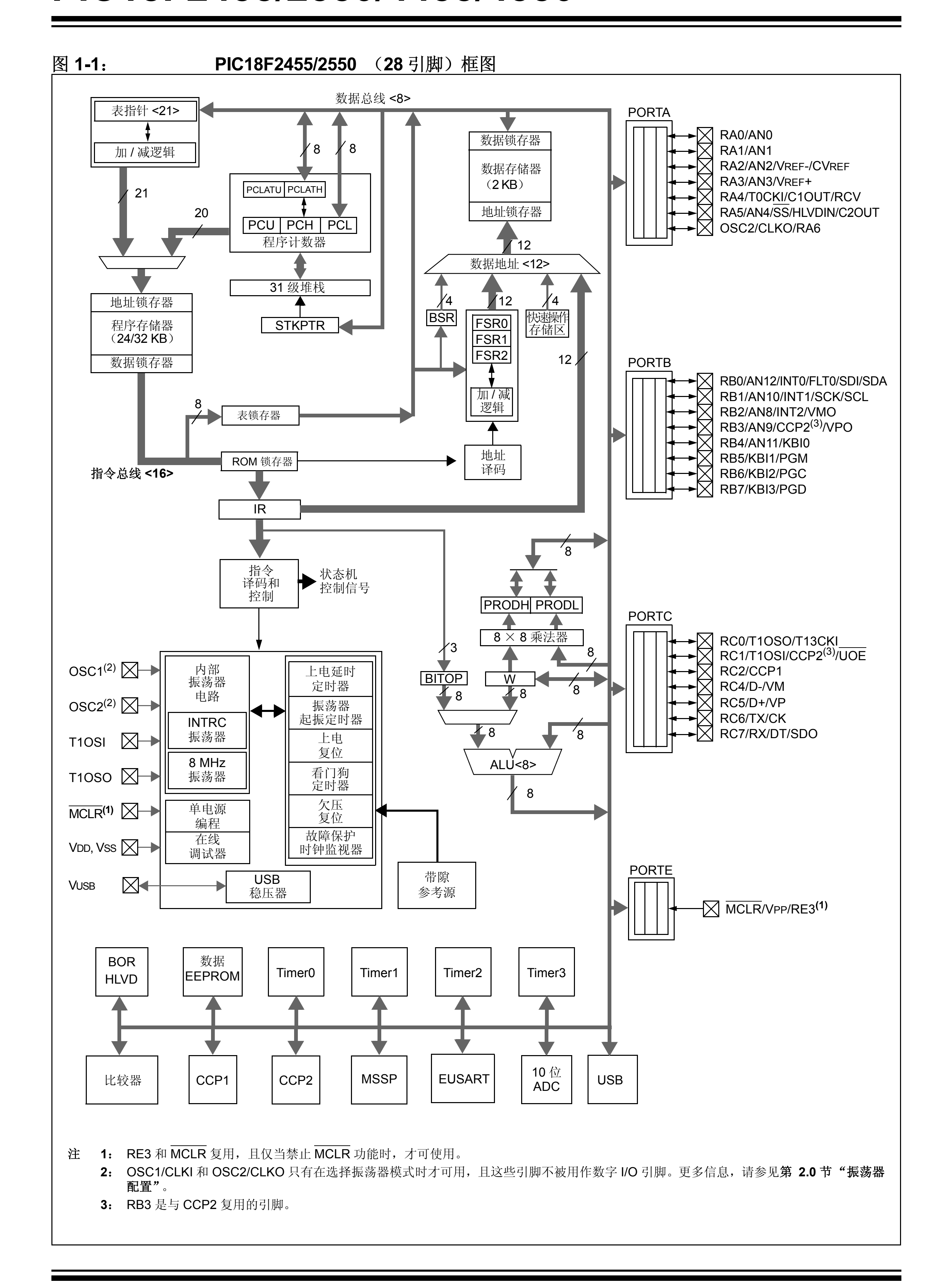
表 1-2 和表 1-3 列举了所有器件的引脚说明。

和所有 Microchip PIC18 器件一样，PIC18F2455/2550/4455/4550 系列的产品也有标准器件和低压器件可供选择。器件编号中标有字母 “F” 的是带有增强型闪存存储器的标准器件 （如 PIC18**F**2550），其工作电压 V_{DD} 范围为 4.2V 到 5.5V。编号中标有 “LF” 的低压器件（如 PIC18**LF**2550）可工作在扩展的 V_{DD} 范围 （2.0V 到 5.5V）中。

表 **1-1**： 器件特性

特性	PIC18F2455	PIC18F2550	PIC18F4455	PIC18F4550
工作频率	DC – 48 MHz	DC – 48 MHz	DC – 48 MHz	DC – 48 MHz
程序存储器 （字节）	24576	32768	24576	32768
程序存储器 （指令）	12288	16384	12288	16384
数据存储器 （字节）	2048	2048	2048	2048
数据 EEPROM 存储器 （字节）	256	256	256	256
中断源	19	19	20	20
I/O 端口	端口 A、 B、 C （和 E）	端口 A、 B、 C （和 E）	端口 A、 B、 C、 D 和 E	端口 A、 B、 C、 D 和 E
定时器	4	4	4	4
捕捉 / 比较 /PWM 模块	2	2	1	1
增强型捕捉 / 比较 /PWM 模块	0	0	1	1
串行通信	MSSP 和 增强型 USART	MSSP 和 增强型 USART	MSSP 和 增强型 USART	MSSP 和 增强型 USART
通用串行总线 （USB）模块	1	1	1	1
并行通信端口 （SPP）	否	否	是	是
10 位模数转换模块	10 路输入通道	10 路输入通道	13 路输入通道	13 路输入通道
比较器	2	2	2	2
复位 （和延迟）	POR、 BOR、 RESET 指令、 堆栈满、 堆栈下溢 （PWRT 和 OST）、 $\overline{\text{MCLR}}$ （可选）和 WDT	POR、 BOR、 RESET 指令、 堆栈满、 堆栈下溢 （PWRT 和 OST）、 $\overline{\text{MCLR}}$ （可选）和 WDT	POR、 BOR、 RESET 指令、 堆栈满、 堆栈下溢 （PWRT 和 OST）、 $\overline{\text{MCLR}}$ （可选）和 WDT	POR、 BOR、 RESET 指令、 堆栈满、 堆栈下溢 （PWRT 和 OST）、 $\overline{\text{MCLR}}$ （可选）和 WDT
可编程低压检测	是	是	是	是
可编程欠压复位	是	是	是	是
指令集	75 条指令； 启用扩展指令集后总共为 83 条指令	75 条指令； 启用扩展指令集后总共为 83 条指令	75 条指令； 启用扩展指令集后总共为 83 条指令	75 条指令； 启用扩展指令集后总共为 83 条指令
封装	28 引脚 PDIP 28 引脚 SOIC	28 引脚 PDIP 28 引脚 SOIC	40 引脚 PDIP 44 引脚 QFN 44 引脚 TQFP	40 引脚 PDIP 44 引脚 QFN 44 引脚 TQFP

图 1-1： PIC18F2455/2550 （28 引脚）框图

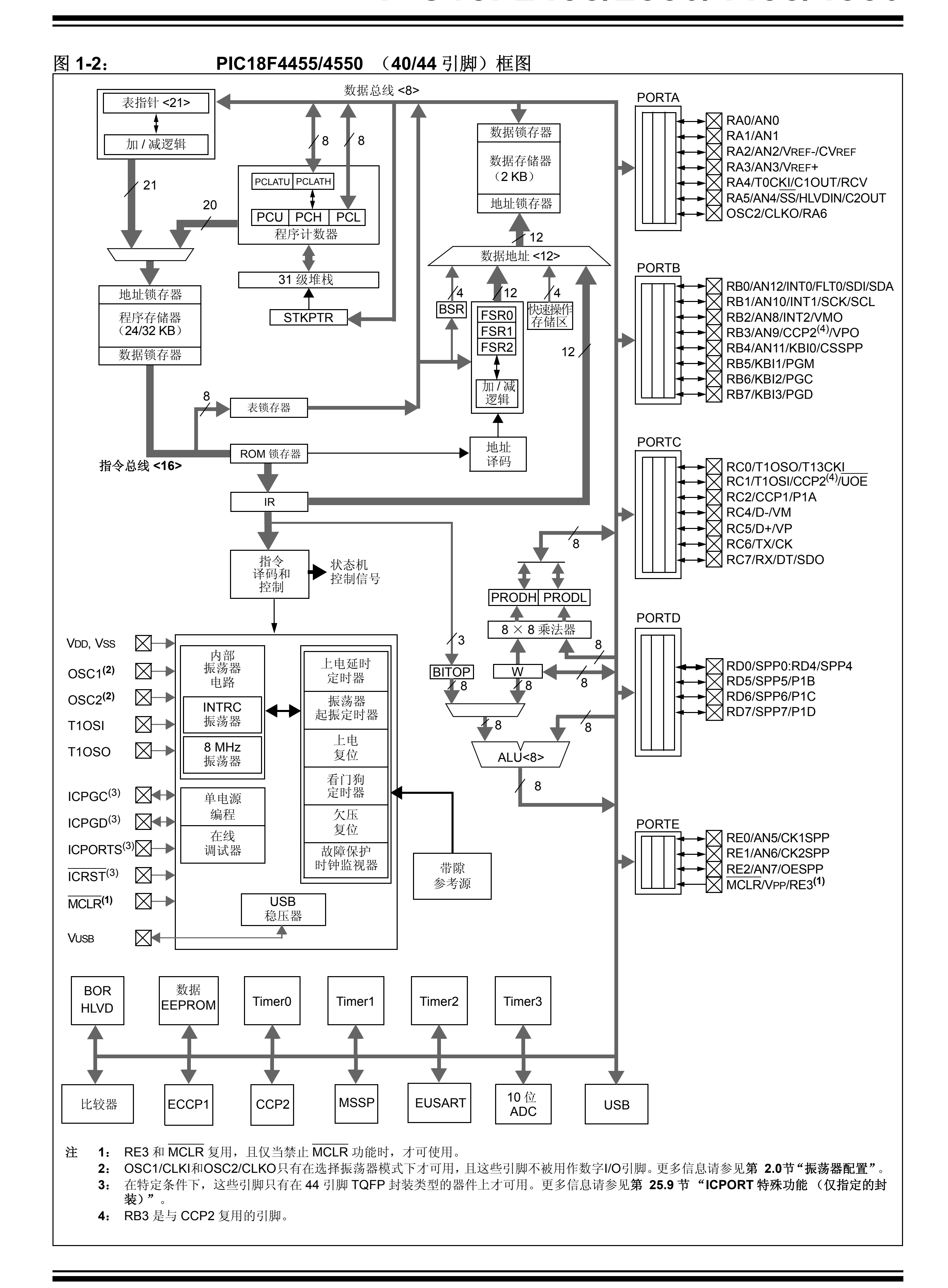


注 1： RE3 和 MCLR 复用，且仅当禁止 MCLR 功能时，才可使用。

2： OSC1/CLKI 和 OSC2/CLKO 只有在选择振荡器模式时才可用，且这些引脚不被用作数字 I/O 引脚。更多信息，请参见**第 2.0 节 “振荡器配置”**。

3： RB3 是与 CCP2 复用的引脚。

图 1-2： PIC18F4455/4550 （40/44 引脚）框图



注 1： RE3 和 MCLR 复用，且仅当禁止 MCLR 功能时，才可使用。

2： OSC1/CLKI和OSC2/CLKO只有在选择振荡器模式下才可用，且这些引脚不被用作数字I/O引脚。更多信息请参见**第 2.0节“振荡器配置”**。

3： 在特定条件下，这些引脚只有在 44 引脚 TQFP 封装类型的器件上才可用。更多信息请参见**第 25.9 节 “ICPORT 特殊功能 （仅指定的封装）”**。

4： RB3 是与 CCP2 复用的引脚。

表 **1-2**： **PIC18F2455/2550 I/O** 引脚说明

引脚名称	引脚号 PDIP 和 SOIC	引脚类型	缓冲器类型	说明
$\overline{MCLR}$/VPP/RE3	1			主复位 （输入）或编程电压 （输入）。
$\overline{MCLR}$		I	ST	主清零 （复位）输入。此引脚为低电平时，器件复位。
VPP		P		编程电压输入。
RE3		I	ST	数字输入。
OSC1/CLKI	9			振荡器晶振或外部时钟输入。
OSC1		I	模拟	振荡器晶振输入或外部时钟源输入。
CLKI		I	模拟	外部时钟源输入。总是与 OSC1 引脚功能复用。（见 OSC2/CLKO 引脚。）
OSC2/CLKO/RA6	10			振荡器晶振或时钟输出。
OSC2		O	—	振荡器晶振输出。在晶振模式下，该引脚与晶振或谐振器相连。
CLKO		O	—	在选定模式下， OSC2 引脚输出 CLKO 信号，该信号是 OSC1 信号的 4 分频信号，其频率等于指令周期的倒数。
RA6		I/O	TTL	通用 I/O 引脚。

图注： TTL = TTL 兼容输入　　CMOS = CMOS 兼容输入或输出
ST = CMOS 电平的施密特触发器输入　　I = 输入
O = 输出　　P = 电源

注 **1**： 当配置位 CCP2MX 被清零时，对 CCP2 进行其他分配。
2： 当配置位 CCP2MX 被置 1 时，对 CCP2 进行默认分配。

表 **1-2：** **PIC18F2455/2550 I/O** 引脚说明 （续）

引脚名称	引脚号 PDIP 和 SOIC	引脚类型	缓冲器类型	说明
				PORTA 是双向 I/O 端口。
RA0/AN0	2			
RA0		I/O	TTL	数字 I/O。
AN0		I	模拟	模拟输入 0。
RA1/AN1	3			
RA1		I/O	TTL	数字 I/O。
AN1		I	模拟	模拟输入 1。
RA2/AN2/VREF-/CVREF	4			
RA2		I/O	TTL	数字 I/O。
AN2		I	模拟	模拟输入 2。
VREF-		I	模拟	A/D 参考电压 （低电平）输入。
CVREF		O	模拟	模拟比较器参考电压输出。
RA3/AN3/VREF+	5			
RA3		I/O	TTL	数字 I/O。
AN3		I	模拟	模拟输入 3。
VREF+		I	模拟	A/D 参考电压 （高电平）输入。
RA4/T0CKI/C1OUT/RCV	6			
RA4		I/O	ST	数字 I/O。
T0CKI		I	ST	Timer0 外部时钟输入。
C1OUT		O	—	比较器 1 输出。
RCV		I	TTL	外部 USB 收发器 RCV 输入。
RA5/AN4/$\overline{SS}$/HLVDIN/C2OUT	7			
RA5		I/O	TTL	数字 I/O。
AN4		I	模拟	模拟输入 4。
$\overline{SS}$		I	TTL	SPI 从动选择输入。
HLVDIN		I	模拟	高 / 低压检测输入。
C2OUT		O	—	比较器 2 输出。
RA6	—	—	—	见 OSC2/CLKO/RA6 引脚信息。

图注： TTL = TTL 兼容输入 　　CMOS = CMOS 兼容输入或输出
ST = CMOS 电平的施密特触发器输入 　　I = 输入
O = 输出 　　P = 电源

注 **1：** 当配置位 CCP2MX 被清零时，对 CCP2 进行其他分配。
2： 当配置位 CCP2MX 被置 1 时，对 CCP2 进行默认分配。

表 1-2： PIC18F2455/2550 I/O 引脚说明 （续）

引脚名称	引脚号 PDIP 和 SOIC	引脚类型	缓冲器类型	说明
				PORTB 是双向 I/O 端口。 PORTB 在所有输入端都可以软件编程为内部弱上拉。
RB0/AN12/INT0/FLT0/SDI/SDA	21			
RB0		I/O	TTL	数字 I/O。
AN12		I	模拟	模拟输入 12。
INT0		I	ST	外部中断 0。
FLT0		I	ST	CCP1 模块 PWM 故障输入。
SDI		I	ST	SPI 数据输入。
SDA		I/O	ST	I^2C™ 数据 I/O。
RB1/AN10/INT1/SCK/SCL	22			
RB1		I/O	TTL	数字 I/O。
AN10		I	模拟	模拟输入 10。
INT1		I	ST	外部中断 1。
SCK		I/O	ST	SPI 模式的同步串行时钟输入 / 输出。
SCL		I/O	ST	I^2C 模式的同步串行时钟输入 / 输出。
RB2/AN8/INT2/VMO	23			
RB2		I/O	TTL	数字 I/O。
AN8		I	模拟	模拟输入 8。
INT2		I	ST	外部中断 2。
VMO		O	—	外部 USB 收发器 VMO 输出。
RB3/AN9/CCP2/VPO	24			
RB3		I/O	TTL	数字 I/O。
AN9		I	模拟	模拟输入 9。
CCP2[1]		I/O	ST	捕捉 2 输入 / 比较 2 输出 /PWM2 输出。
VPO		O	—	外部 USB 收发器 VPO 输出。
RB4/AN11/KBI0	25			
RB4		I/O	TTL	数字 I/O。
AN11		I	模拟	模拟输入 11。
KBI0		I	TTL	电平变化中断引脚。
RB5/KBI1/PGM	26			
RB5		I/O	TTL	数字 I/O。
KBI1		I	TTL	电平变化中断引脚。
PGM		I/O	ST	低压 ICSP™ 编程使能引脚。
RB6/KBI2/PGC	27			
RB6		I/O	TTL	数字 I/O。
KBI2		I	TTL	电平变化中断引脚。
PGC		I/O	ST	在线调试器和 ICSP 编程时钟引脚。
RB7/KBI3/PGD	28			
RB7		I/O	TTL	数字 I/O。
KBI3		I	TTL	电平变化中断引脚。
PGD		I/O	ST	在线调试器和 ICSP 编程数据引脚。

图注： TTL = TTL 兼容输入　　CMOS = CMOS 兼容输入或输出
ST = CMOS 电平的施密特触发器输入　　I = 输入
O = 输出　　P = 电源

注 1： 当配置位 CCP2MX 被清零时，对 CCP2 进行其他分配。
2： 当配置位 CCP2MX 被置 1 时，对 CCP2 进行默认分配。

表 **1-2：** **PIC18F2455/2550 I/O** 引脚说明 （续）

引脚名称	引脚号 PDIP 和 SOIC	引脚类型	缓冲器类型	说明
				PORTC 是双向 I/O 端口。
RC0/T1OSO/T13CKI	11			
RC0		I/O	ST	数字 I/O。
T1OSO		O	—	Timer1 振荡器输出。
T13CKI		I	ST	Timer1/Timer3 外部时钟输入。
RC1/T1OSI/CCP2/UOE	12			
RC1		I/O	ST	数字 I/O。
T1OSI		I	CMOS	Timer1 振荡器输入。
CCP2(2)		I/O	ST	捕捉 2 输入 / 比较 2 输出 /PWM 2 输出。
UOE		—	—	外部 USB 收发器 OE 输出。
RC2/CCP1	13			
RC2		I/O	ST	数字 I/O。
CCP1		I/O	ST	捕捉 1 输入 / 比较 1 输出 /PWM 1 输出。
RC4/D-/VM	15			
RC4		I	TTL	数字输入。
D-		I/O	—	USB 差分负信号线 （输入 / 输出）。
VM		I	TTL	外部 USB 收发器 VM 输入。
RC5/D+/VP	16			
RC5		I	TTL	数字输入。
D+		I/O	—	USB 差分正信号线 （输入 / 输出）。
VP		O	TTL	外部 USB 收发器 VP 输入。
RC6/TX/CK	17			
RC6		I/O	ST	数字 I/O。
TX		O	—	EUSART 异步发送。
CK		I/O	ST	EUSART 同步时钟 （见 RX/DT 引脚）。
RC7/RX/DT/SDO	18			
RC7		I/O	ST	数字 I/O。
RX		I	ST	EUSART 异步接收。
DT		I/O	ST	EUSART 同步数据 （见 TX/CK 引脚）。
SDO		O	—	SPI 数据输出。
RE3	—	—	—	见 MCLR/VPP/RE3 引脚。
VUSB	14	O	—	内部 USB 3.3V 稳压器
VSS	8, 19	P	—	逻辑电路和 I/O 引脚的参考地。
VDD	20	P	—	逻辑电路和 I/O 引脚的电源正极。

图注：
TTL = TTL 兼容输入　　CMOS = CMOS 兼容输入或输出
ST = CMOS 电平的施密特触发器输入　　I = 输入
O = 输出　　P = 电源

注 **1：** 当配置位 CCP2MX 被清零时，对 CCP2 进行其他分配。
2： 当配置位 CCP2MX 被置 1 时，对 CCP2 进行默认分配。

表 1-3： PIC18F4455/4550 I/O 引脚说明

引脚名称	引脚号			引脚类型	缓冲器类型	说明
	PDIP	QFN	TQFP			
$\overline{MCLR}$/VPP/RE3	1	18	18			主清零 （输入）或编程电压 （输入）。
MCLR				I	ST	主清零 （复位）输入。该引脚为低电平时，器件复位。
VPP				P		编程电压输入。
RE3				I	ST	数字输入。
OSC1/CLKI	13	32	30			振荡器晶振或外部时钟输入。
OSC1				I	模拟	振荡器晶振输入或外部时钟源输入。
CLKI				I	模拟	外部时钟源输入。总是与 OSC1 引脚功能复用。（见 OSC2/CLKO 引脚。）
OSC2/CLKO/RA6	14	33	31			振荡器晶振或时钟输出。
OSC2				O	—	振荡器晶振输出。在晶振模式下，该引脚与晶振或谐振器相连。
CLKO				O	—	在 RC 模式下， OSC2 引脚输出 CLKO 信号，该信号是OSC1 信号的 4 分频信号，频率等于指令周期的倒数。
RA6				I/O	TTL	通用 I/O 引脚。

图注： TTL = TTL 兼容输入　　CMOS = CMOS 兼容输入或输出
ST = CMOS 电平的施密特触发器输入　　I = 输入
O = 输出　　P = 电源

注 **1：** 当配置位 CCP2MX 被清零时，对 CCP2 进行其他分配。

2： 当配置位 CCP2MX 被置 1 时，对 CCP2 进行默认分配。

3： 除非 $\overline{ICPRT}$ 配置位置 1，否则这些引脚都处于悬空状态。对于 NC/ICPORTS，除非 ICPRT 被置 1 且 $\overline{DEBUG}$ 配置位被清零，否则该引脚处于悬空状态。

表 **1-3**： **PIC18F4455/4550 I/O** 引脚说明 （续）

引脚名称	引脚号			引脚类型	缓冲器类型	说明
	PDIP	QFN	TQFP			
						PORTA 是双向 I/O 端口。
RA0/AN0	2	19	19			
RA0				I/O	TTL	数字 I/O。
AN0				I	模拟	模拟输入 0。
RA1/AN1	3	20	20			
RA1				I/O	TTL	数字 I/O。
AN1				I	模拟	模拟输入 1。
RA2/AN2/VREF-/CVREF	4	21	21			
RA2				I/O	TTL	数字 I/O。
AN2				I	模拟	模拟输入 2。
VREF-				I	模拟	A/D 参考电压 （低电平）输入。
CVREF				O	模拟	模拟比较器参考电压输出。
RA3/AN3/VREF+	5	22	22			
RA3				I/O	TTL	数字 I/O。
AN3				I	模拟	模拟输入 3。
VREF+				I	模拟	A/D 参考电压 （高电平）输入。
RA4/T0CKI/C1OUT/RCV	6	23	23			
RA4				I/O	ST	数字 I/O。
T0CKI				I	ST	Timer0 外部时钟源输入。
C1OUT				O	—	比较器 1 输出。
RCV				I	TTL	外部 USB 收发器 RCV 输入。
RA5/AN4/$\overline{SS}$/HLVDIN/C2OUT	7	24	24			
RA5				I/O	TTL	数字 I/O。
AN4				I	模拟	模拟输入 4。
$\overline{SS}$				I	TTL	SPI 从动选择输入。
HLVDIN				I	模拟	高 / 低压检测输入。
C2OUT				O	—	比较器 2 输出。
RA6	—	—	—	—	—	见 OSC2/CLKO/RA6 引脚。

图注： TTL = TTL 兼容输入　　CMOS = CMOS 兼容输入或输出
ST = CMOS 电平的施密特触发器输入　　I = 输入
O = 输出　　P = 电源

注 **1**： 当配置位 CCP2MX 被清零时，对 CCP2 进行其他分配。
2： 当配置位 CCP2MX 被置 1 时，对 CCP2 进行默认分配。
3： 除非 ICPRT 配置位置 1，否则这些引脚都处于悬空状态。对于 NC/ICPORTS，除非 ICPRT 被置 1 且 $\overline{DEBUG}$ 配置位被清零，否则该引脚处于悬空状态。

表 1-3： PIC18F4455/4550 I/O 引脚说明 （续）

引脚名称	引脚号			引脚类型	缓冲器类型	说明
	PDIP	QFN	TQFP			
						PORTB 是双向 I/O 端口。 PORTB 的所有输入端都可以软件编程为内部弱上拉。
RB0/AN12/INT0/FLT0/SDI/SDA	33	9	8			
RB0				I/O	TTL	数字 I/O。
AN12				I	模拟	模拟输入 12。
INT0				I	ST	外部中断 0。
FLT0				I	ST	增强型 ECCP1 模块 PWM 故障输入。
SDI				I	ST	SPI 数据输入。
SDA				I/O	ST	I²C™ 数据 I/O。
RB1/AN10/INT1/SCK/SCL	34	10	9			
RB1				I/O	TTL	数字 I/O。
AN10				I	模拟	模拟输入 10。
INT1				I	ST	外部中断 1。
SCK				I/O	ST	SPI 模式的同步串行时钟输入 / 输出。
SCL				I/O	ST	I²C 模式的同步串行时钟输入 / 输出。
RB2/AN8/INT2/VMO	35	11	10			
RB2				I/O	TTL	数字 I/O。
AN8				I	模拟	模拟输入 8。
INT2				I	ST	外部中断 2。
VMO				O	—	外部 USB 收发器 VMO 输出。
RB3/AN9/CCP2/VPO	36	12	11			
RB3				I/O	TTL	数字 I/O。
AN9				I	模拟	模拟输入 9。
CCP2(1)				I/O	ST	捕捉 2 输入 / 比较 2 输出 /PWM 2 输出。
VPO				O	—	外部 USB 收发器 VPO 输出。
RB4/AN11/KBI0/CSSPP	37	14	14			
RB4				I/O	TTL	数字 I/O。
AN11				I	模拟	模拟输入 11。
KBI0				I	TTL	电平变化中断引脚。
CSSPP				O	—	SPP 片选控制输出。
RB5/KBI1/PGM	38	15	15			
RB5				I/O	TTL	数字 I/O。
KBI1				I	TTL	电平变化中断引脚。
PGM				I/O	ST	低压 ICSP™ 编程使能引脚。
RB6/KBI2/PGC	39	16	16			
RB6				I/O	TTL	数字 I/O。
KBI2				I	TTL	电平变化中断引脚。
PGC				I/O	ST	在线调试器和 ICSP 编程时钟引脚。
RB7/KBI3/PGD	40	17	17			
RB7				I/O	TTL	数字 I/O。
KBI3				I	TTL	电平变化中断引脚。
PGD				I/O	ST	在线调试器和 ICSP 编程数据引脚。

图注： TTL = TTL 兼容输入　　CMOS = CMOS 兼容输入或输出
ST = CMOS 电平的施密特触发器输入　　I = 输入
O = 输出　　P = 电源

注 1： 当配置位 CCP2MX 被清零时，对 CCP2 进行其他分配。

2： 当配置位 CCP2MX 被置 1 时，对 CCP2 进行默认分配。

3： 除非 ICPRT 配置位置 1，否则这些引脚都处于悬空状态。对于 NC/ICPORTS，除非 ICPRT 被置 1 且 DEBUG 配置位被清零，否则该引脚处于悬空状态。

表 **1-3：** **PIC18F4455/4550 I/O** 引脚说明 （续）

引脚名称	引脚号			引脚类型	缓冲器类型	说明
	PDIP	**QFN**	**TQFP**			
						PORTC 是双向 I/O 端口。
RC0/T1OSO/T13CKI	15	34	32			
RC0				I/O	ST	数字 I/O。
T1OSO				O	—	Timer1 振荡器输出。
T13CKI				I	ST	Timer1/Timer3 外部时钟输入。
RC1/T1OSI/CCP2/UOE	16	35	35			
RC1				I/O	ST	数字 I/O。
T1OSI				I	CMOS	Timer1 振荡器输入。
CCP2[2]				I/O	ST	捕捉 2 输入 / 比较 2 输出 /PWM 2 输出。
UOE				O	—	外部 USB 收发器 OE 输出。
RC2/CCP1/P1A	17	36	36			
RC2				I/O	ST	数字 I/O。
CCP1				I/O	ST	捕捉 1 输入 / 比较 1 输出 /PWM 1 输出。
P1A				O	TTL	增强型 CCP1 PWM 输出，通道 A。
RC4/D-/VM	23	42	42			
RC4				I	TTL	数字输入。
D-				I/O	—	USB 差分负信号线 （输入 / 输出）。
VM				I	TTL	外部 USB 收发器 VM 输入。
RC5/D+/VP	24	43	43			
RC5				I	TTL	数字输入。
D+				I/O	—	USB 差分正信号线 （输入 / 输出）。
VP				I	TTL	外部 USB 收发器 VP 输入。
RC6/TX/CK	25	44	44			
RC6				I/O	ST	数字 I/O。
TX				O	—	EUSART 异步发送。
CK				I/O	ST	EUSART 同步时钟 （见 RX/DT 引脚）。
RC7/RX/DT/SDO	26	1	1			
RC7				I/O	ST	数字 I/O。
RX				I	ST	EUSART 异步接收。
DT				I/O	ST	EUSART 同步数据 （见 TX/CK 引脚）。
SDO				O	—	SPI 数据输出。

图注： TTL = TTL 兼容输入　　CMOS = CMOS 兼容输入或输出
ST = CMOS 电平的施密特触发器输入　　I = 输入
O = 输出　　P = 电源

注 **1：** 当配置位 CCP2MX 被清零时，对 CCP2 进行其他分配。

2： 当配置位 CCP2MX 被置 1 时，对 CCP2 进行默认分配。

3： 除非 ICPRT 配置位置 1，否则这些引脚都处于悬空状态。对于 NC/ICPORTS，除非 ICPRT 被置 1 且 DEBUG 配置位被清零，否则该引脚处于悬空状态。

　初稿　

表 1-3： PIC18F4455/4550 I/O 引脚说明 （续）

引脚名称	引脚号			引脚类型	缓冲器类型	说明
	PDIP	QFN	TQFP			
						PORTD 是双向 I/O 端口或并行通信端口 （SPP）。当使能 SPP 模块时，这些引脚具有 TTL 输入缓冲器。
RD0/SPP0	19	38	38			
RD0				I/O	ST	数字 I/O。
SPP0				I/O	TTL	并行通信端口数据。
RD1/SPP1	20	39	39			
RD1				I/O	ST	数字 I/O。
SPP1				I/O	TTL	并行通信端口数据。
RD2/SPP2	21	40	40			
RD2				I/O	ST	数字 I/O。
SPP2				I/O	TTL	并行通信端口数据。
RD3/SPP3	22	41	41			
RD3				I/O	ST	数字 I/O。
SPP3				I/O	TTL	并行通信端口数据。
RD4/SPP4	27	2	2			
RD4				I/O	ST	数字 I/O。
SPP4				I/O	TTL	并行通信端口数据。
RD5/SPP5/P1B	28	3	3			
RD5				I/O	ST	数字 I/O。
SPP5				I/O	TTL	并行通信端口数据。
P1B				O	—	增强型 CCP1 PWM 输出，通道 B。
RD6/SPP6/P1C	29	4	4			
RD6				I/O	ST	数字 I/O。
SPP6				I/O	TTL	并行通信端口数据。
P1C				O	—	增强型 CCP1 PWM 输出，通道 C。
RD7/SPP7/P1D	30	5	5			
RD7				I/O	ST	数字 I/O。
SPP7				I/O	TTL	并行通信端口数据。
P1D				O	—	增强型 CCP1 PWM 输出，通道 D。

图注： TTL = TTL 兼容输入　　CMOS = CMOS 兼容输入或输出
ST = CMOS 电平的施密特触发器输入　　I = 输入
O = 输出　　P = 电源

注 1： 当配置位 CCP2MX 被清零时，对 CCP2 进行其他分配。
2： 当配置位 CCP2MX 被置 1 时，对 CCP2 进行默认分配。
3： 除非 ICPRT 配置位置 1，否则这些引脚都处于悬空状态。对于 NC/ICPORTS，除非 ICPRT 被置 1 且 $\overline{\text{DEBUG}}$ 配置位被清零，否则该引脚处于悬空状态。

 初稿

表 **1-3**： **PIC18F4455/4550 I/O** 引脚说明 （续）

引脚名称	引脚号 PDIP	引脚号 QFN	引脚号 TQFP	引脚类型	缓冲器类型	说明
						PORTE 是双向 I/O 端口。
RE0/AN5/CK1SPP	8	25	25			
RE0				I/O	ST	数字 I/O。
AN5				I	模拟	模拟输入 5。
CK1SPP				O	—	SPP 时钟 1 输出。
RE1/AN6/CK2SPP	9	26	26			
RE1				I/O	ST	数字 I/O。
AN6				I	模拟	模拟输入 6。
CK2SPP				O	—	SPP 时钟 2 输出。
RE2/AN7/OESPP	10	27	27			
RE2				I/O	ST	数字 I/O。
AN7				I	模拟	模拟输入 7。
OESPP				O	—	SPP 输出使能输出。
RE3	—	—	—	—	—	见 $\overline{\text{MCLR}}$/VPP/RE3 引脚。
VSS	12, 31	6, 30, 31	6, 29	P	—	逻辑电路和 I/O 引脚的参考地。
VDD	11, 32	7, 8, 28, 29	7, 28	P	—	逻辑电路和 I/O 引脚的电源正极。
VUSB	18	37	37	O	—	内部 USB 3.3V 稳压器输出。
NC/ICCK/ICPGC	—	—	12			无连接或专用 ICD/ICSP™ 端口时钟。(3)
ICCK				I/O	ST	在线调试器时钟。
ICPGC				I/O	ST	ICSP 编程时钟。
NC/ICDT/ICPGD	—	—	13			无连接或专用 ICD/ICSP 端口时钟。(3)
ICDT				I/O	ST	在线调试器数据。
ICPGD				I/O	ST	ICSP 编程数据。
NC/$\overline{\text{ICRST}}$/ICVPP	—	—	33			无连接或专用 ICD/ICSP 端口复位信号。(3)
$\overline{\text{ICRST}}$				I	—	主清零 （复位）输入。
ICVPP				P	—	编程电压输入。
NC/ICPORTS	—	—	34	P	—	无连接或 28 引脚器件仿真信号。(3)
ICPORTS						当连接到 Vss 时，使能 28 引脚器件仿真。
NC	—	13	—	—	—	无连接。

图注： TTL = TTL 兼容输入　　CMOS = CMOS 兼容输入或输出
ST = CMOS 电平的施密特触发器输入　　I = 输入
O = 输出　　P = 电源

注 **1**： 当配置位 CCP2MX 被清零时，对 CCP2 进行其他分配。
2： 当配置位 CCP2MX 被置 1 时，对 CCP2 进行默认分配。
3： 除非 ICPRT 配置位置 1，否则这些引脚都处于悬空状态。对于 NC/ICPORTS，除非 ICPRT 被置 1 且 $\overline{\text{DEBUG}}$ 配置位被清零，否则该引脚处于悬空状态。

注：

2.0 振荡器配置

2.1 概述

PIC18F2455/2550/4455/4550 系列器件采用的振荡器和单片机时钟系统与以前的 PIC18F 系列器件不同。由于增加的 USB 模块对时钟源的稳定性有独特要求，所以必须提供一个符合 USB 低速和全速规范的独立时钟源。

为了适应这些要求，PIC18F2455/2550/4455/4550 器件采用了一个新的时钟，由它来提供 48 MHz 的时钟以进行全速 USB 操作。由于该时钟由主时钟源驱动，故增加了一个额外的预分频器和后分频器系统以提供宽范围的振荡频率。图 2-1 概述了振荡器的构成。

在 PIC18 增强型单片机中所使用的其他振荡器功能（如内部振荡电路和时钟切换）保持不变。在本章后面的部分将讨论这些功能。

2.1.1 振荡器控制

PIC18F2455/2550/4455/4550 器件的振荡器操作通过两个配置寄存器和两个控制寄存器控制。配置寄存器，CONFIG1L 和 CONFIG1H，用于选择振荡模式和 USB 预分频器 / 后分频器选项。在对器件编程时设置这两个配置寄存器中的配置位，这些位将保留此配置直到重新编程为止。

OSCCON 寄存器 （寄存器 2-2）用于选择工作时钟模式；它主要用于控制功耗管理模式下的时钟切换。在第 **2.4.1 节“振荡器控制寄存器”**中讨论了它的使用。

OSCTUNE 寄存器 （寄存器 2-1）用来调整 INTRC 频率源，以及选择驱动几个特殊功能的低频时钟源。在第 **2.2.5.2 节“OSCTUNE寄存器”**中讨论了它的使用。

2.2 振荡器类型

PIC18F2455/2550/4455/4550 器件可以在 12 种不同的振荡模式下运行。与以前的 PIC18 增强型单片机不同，其中 4 种振荡模式将同时使用两种类型的振荡器。用户可以通过设置FOSC3:FOSC0配置位来选择以下任一模式：

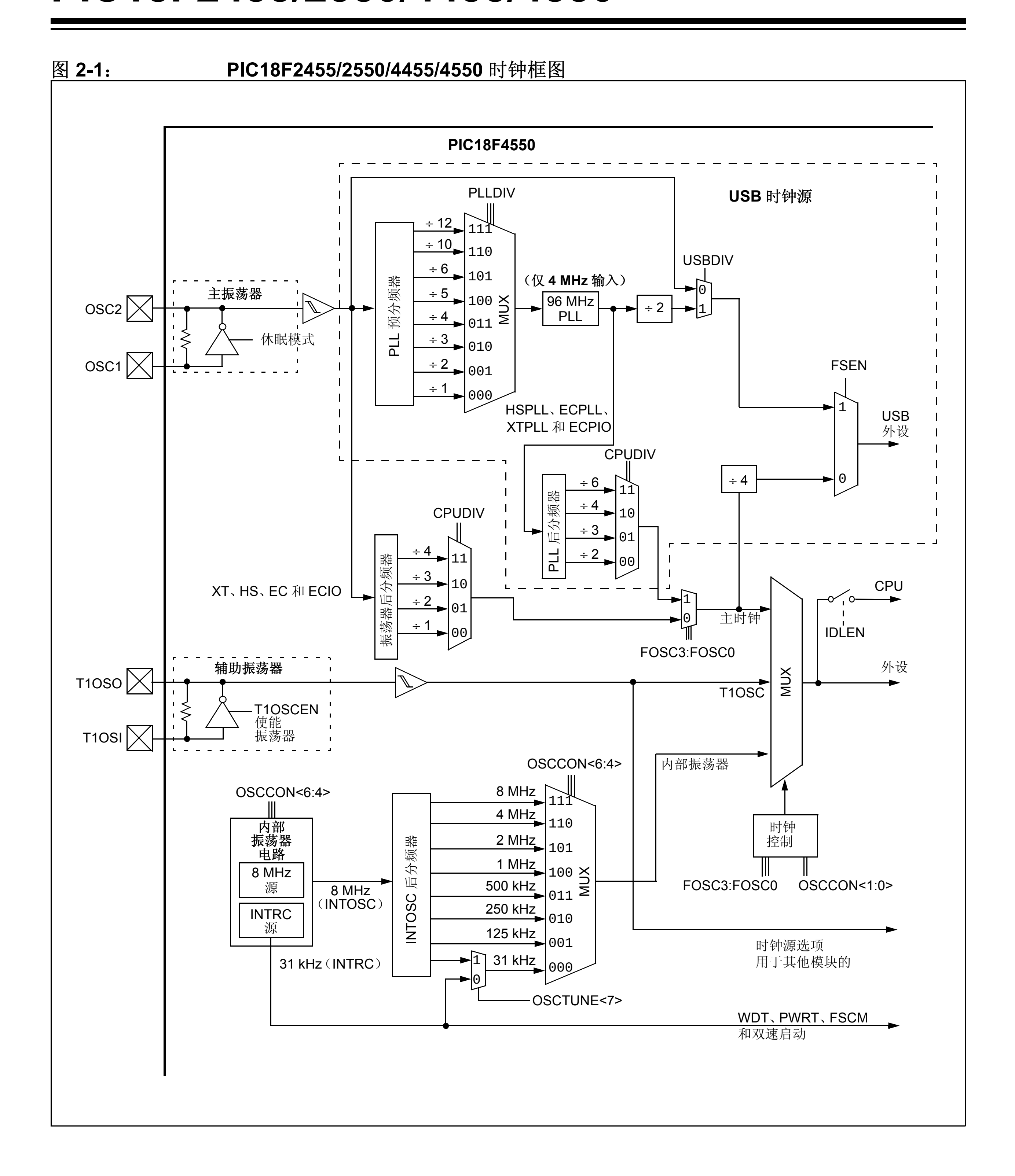
1. XT 晶振 / 谐振器模式
2. XTPLL 使能 PLL 的晶振 / 谐振器模式
3. HS 高速晶振 / 谐振器模式
4. HSPLL 使能 PLL 的高速晶振 / 谐振器模式
5. EC 带 Fosc/4 输出的外部时钟模式
6. ECIO RA6 作为 I/O 引脚的外部时钟模式
7. ECPLL 使能 PLL 并通过 RA6 输出 Fosc/4 信号的外部时钟模式
8. ECPIO 使能 PLL 并将 RA6 作为 I/O 引脚的外部时钟模式
9. INTHS 内部振荡器用作单片机时钟源， HS 振荡器用作 USB 时钟源
10. INTXT 内部振荡器用作单片机时钟源， XT 振荡器用作 USB 时钟源
11. INTIO 内部振荡器用作 单片机时钟源， EC 振荡器用作 USB 时钟源， RA6 用作数字 I/O 引脚
12. INTCKO 内部振荡器用作单片机时钟源， EC 振荡器用作 USB 时钟源，通过 RA6 引脚输出 Fosc/4 信号

2.2.1 振荡模式和 USB 操作

由于 USB 模块的独特要求，所以必须使用不同的时钟操作方法。在以前的 PICmicro® 器件中，内核和外设的时钟信号都由一个振荡器源提供，通常时钟源是主振荡器、辅助振荡器或内部振荡器。对于 PIC18F2455/2550/4455/4550 器件，主振荡器成为了 USB 模块的一部分，不能与任何其他的时钟源替换使用。因此，USB 模块必须由主时钟源提供时钟信号；然而，单片机内核和其他外设仍可由辅助振荡器或内部振荡器独立地提供时钟信号。

由于 USB 的定时要求，当使能 USB 模块时，需要使用 6 MHz 或 48 MHz 的内部时钟。幸运地是，当使用主振荡器时，单片机和其他外设不需要以此时钟速度运行。有多种方法能达到 USB 模块的时钟要求，并且主振荡器仍能灵活地为器件的其他部分提供时钟信号。在第 **2.3 节 “USB 的振荡器设置”**中对这些进行了详细讨论。

图 2-1： PIC18F2455/2550/4455/4550 时钟框图



 初稿

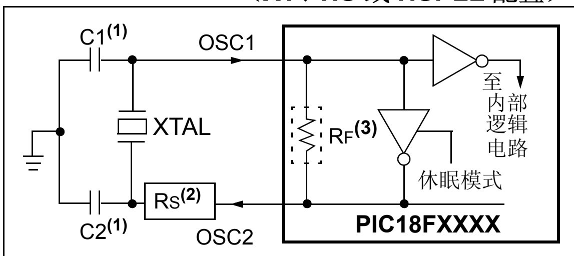
2.2.2 晶振 / 陶瓷谐振器

在 HS、HSPLL、XT 和 XTPLL 振荡器模式下，晶振或陶瓷谐振器与 OSC1 和 OSC2 引脚连接来产生振荡信号。图 2-2 显示了引脚连接方式。

振荡器的设计要求使用平行切割晶体。

注：	使用顺序切割的晶体，可能会使振荡器产生的频率不在晶体制造厂商所给的参数范围内。

图 2-2： 晶振 / 陶瓷谐振器工作原理（XT、HS 或 HSPLL 配置）



注 1：如需了解 C1 和 C2 的初始值，请参见表 2-1 和表 2-2。
2：AT 条形切割的晶体可能会需要一个串联电阻（RS）。
3：RF 的值根据所选择的振荡模式而定。

表 2-1： 陶瓷谐振器的电容选择

使用的典型电容值：			
模式	频率	**OSC1**	**OSC2**
XT	4.0 MHz	33 pF	33 pF
HS	8.0 MHz 16.0 MHz	27 pF 22 pF	27 pF 22 pF

上述电容值仅供设计参考。

已测试这些电容搭配下列谐振器时的基本起振和工作情况。**这些值不是最佳值。**

要得到合适的振荡器工作状况，可能需要不同的电容值。用户应在应用的预期 VDD 和温度范围内测试振荡器的性能。

欲知更多信息，请参见表 2-2 后的“注”。

所使用的谐振器：
4.0 MHz
8.0 MHz
16.0 MHz

表 2-2： 晶振的电容选择

振荡器类型	晶振频率	已测试的典型电容值：	
		C1	**C2**
XT	4 MHz	27 pF	27 pF
HS	4 MHz	27 pF	27 pF
	8 MHz	22 pF	22 pF
	20 MHz	15 pF	15 pF

上述电容值仅供设计参考。

已测试这些电容搭配下列晶振时的基本起振和工作情况。**这些值不是最佳值。**

要得到合适的振荡器工作状况，可能需要不同的电容值。用户应在应用的预期 VDD 和温度范围内测试振荡器的性能。

欲知更多信息，请参见本表后的“注”。

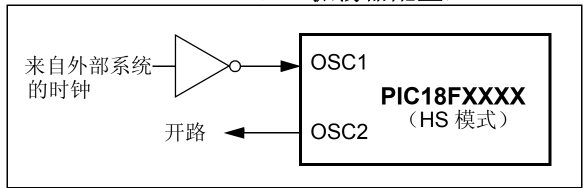
所使用的晶振：
4 MHz
8 MHz
20 MHz

注 1：较高的电容值可以提高振荡器的稳定性，但同时也会增加起振时间。
2：当工作电压 VDD 低于 3V，或使用某些陶瓷谐振器时，可能需要使用 HS 振荡模式或改用晶振。
3：因为每种谐振器 / 晶振都有其自身特性，用户应当向谐振器 / 晶振制造厂商询问外部元件的相应值。
4：为避免对低驱动电平规格的晶体造成过驱动，可能会需要使用电阻 RS。
5：请在应用中的预期 VDD 和温度范围内验证振荡器的性能。

内部后分频器使用户能选择与晶振或谐振器的频率不同的时钟。分频比由 CPUDIV 配置位决定。用户可以选择振荡器频率，或振荡器频率的 1/2、1/3 或 1/4 作为时钟频率。

当单片机处于HS振荡器模式时，也可以使用外部时钟。在这种情况下， OSC2/CLKO 引脚保持开路状态（图 2-3）。

图 2-3： 外部时钟输入连接方式（HS 振荡器配置）

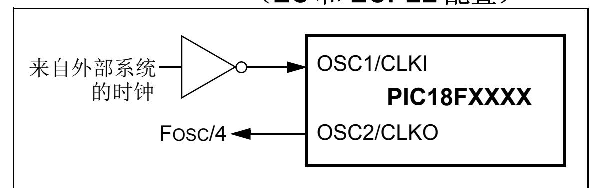


2.2.3 外部时钟输入

EC、ECIO、ECPLL 和 ECPIO 振荡器模式要求 OSC1 引脚与一个外部时钟源相连。在上电复位后或从休眠模式退出后，不需要振荡器起振时间。

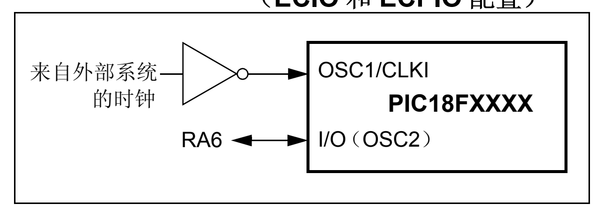
在 EC 和 ECPLL 振荡器模式下，振荡器频率的 4 分频信号由 OSC2 引脚输出。此信号可用于测试或同步其他逻辑。图 2-4 显示了 EC 振荡器模式的引脚连接方式。

图 2-4： 外部时钟输入连接方式（EC 和 ECPLL 配置）



除了 OSC2 引脚成为额外的通用 I/O 引脚之外， ECIO 和 ECPIO 振荡模式与 EC 和 ECPLL 模式的工作方式相同。该 I/O 引脚成为 PORTA 的 bit 6 （RA6）。图 2-5 显示了 ECIO 振荡器模式下的引脚连接方式。

图 2-5： 外部时钟输入连接方式（ECIO 和 ECPIO 配置）



在 XT 和 HS 模式下用于降低时钟频率的内部后分频器在 EC 和 ECIO 模式下同样可用。

2.2.4 PLL 倍频器

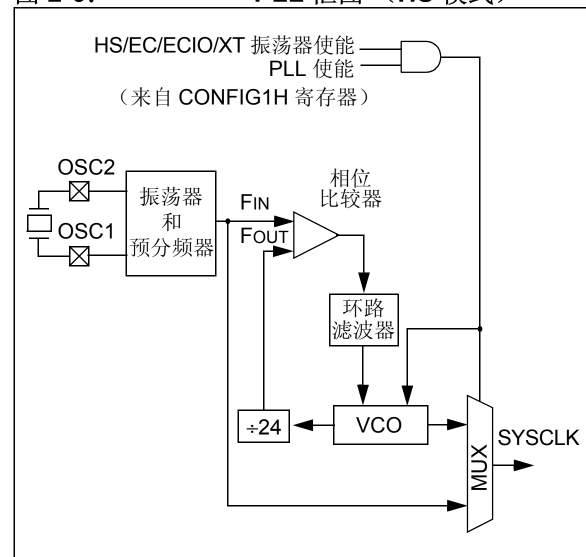
PIC18F2455/2550/4255/4550 器件包括一个锁相环（Phase Locked Loop，PLL）电路。这是专门为使用低速振荡器的 USB 应用提供的，该电路还可被用作单片机时钟源。

在 HSPLL、XTPLL、ECPLL 和 ECPIO 振荡器模式下使能 PLL。PLL 旨在由 4 MHz 的固定输入产生 96 MHz 的固定参考时钟（输出），然后可对该输出进行分频为 USB 和单片机内核提供时钟信号。由于 PLL 有固定频率的输入和输出，所以有 8 个预分频选项以产生到 PLL 的振荡器输入频率。

还有一个独立的后分频器选项可以对PLL输出信号进行分频，然后将分频信号作为单片机的时钟信号。这使 USB 外设和单片机可以使用相同的振荡器输入，但仍然能够以不同的时钟速度工作。与 XT、HS 和 EC 模式的后分频器不同， PLL 输出的可用后分频选项是 2 分频、3 分频、 4 分频和 6 分频。

HSPLL、 ECPLL 和 ECPIO 模式利用 HS 模式振荡器，可产生的频率最高可达 48 MHz。预分频器至多可以对振荡器输入进行 12 分频，以产生 4 MHz 的信号来驱动 PLL。XTPLL 模式只能使用 4 MHz 的输入频率，由它直接驱动 PLL。

图 2-6： PLL 框图 （HS 模式）



 初稿

2.2.5 内部振荡器电路

PIC18F2455/2550/4455/4550 器件包含可产生两种不同时钟信号的内部振荡器电路。两种信号都可以充当单片机的时钟源。如果未使用 USB 外设，内部振荡器可以不使用 OSC1 和 / 或 OSC2 引脚上的外部振荡器电路。

主输出（INTOSC）是一个 8 MHz 的时钟源，可以直接驱动器件时钟。它还可以驱动 INTOSC 后分频器，该后分频器可以提供从 31 kHz 到 4 MHz 的时钟频率。当选择了 125 kHz 到 8 MHz 的时钟频率时，会使能 INTOSC 输出。

另一个时钟源是内部 RC 振荡器 （INTRC），它提供了标称值为 31 kHz 的输出。如果选择 INTRC 作为器件的时钟源，它就会被使能；当使能以下任一功能时，也将自动使能 INTRC：

- 上电延时定时器
- 故障保护时钟监视器
- 看门狗定时器
- 双速启动

第 **25.0 节 "CPU 的特殊性能"** 中详细讨论了这些功能。

通过配置 OSCCON 寄存器（第 32 页）的 IRCF 位，可以选择时钟源频率 （INTOSC 直接频率、 INTRC 直接频率或 INTOSC 后分频器频率）。

2.2.5.1 内部振荡器模式

当内部振荡器用作单片机时钟源时，需要将其他某种振荡模式 （外部时钟或外部晶振 / 谐振器）用作 USB 时钟源。USB 时钟源的选择由某种特定的内部振荡器模式决定。

有 4 种不同的模式可用：

1. INTHS 模式 ：USB 时钟由处于 HS 模式的振荡器提供。
2. INTXT 模式：USB 时钟由处于 XT 模式的振荡器提供。
3. INTCKO 模式：USB 时钟由 OSC1/CLKI 引脚上的外部时钟提供； OSC2/CLKO 引脚输出 Fosc/4。
4. INTIO 模式：USB 时钟由 OSC1/CLKI 引脚上的外部时钟提供； OSC2/CLKO 引脚充当数字 I/O （RA6）。

在这四种模式中，只有 INTIO 模式可以空出一个额外的引脚 （OSC2/CLKO/RA6）用作 I/O 端口引脚。

2.2.5.2 OSCTUNE 寄存器

内部振荡器的输出已在出厂前经过校准，但用户仍可在应用中进行调节，这可以通过写 OSCTUNE 寄存器（寄存器 2-1）来实现。 调节灵敏度在调节范围内保持不变。

当修改了 OSCTUNE 寄存器后，INTOSC 和 INTRC 频率将改变为新的频率。 INTRC 时钟将在 8 个时钟周期（大约$8 \times 32\ \mu s = 256\ \mu s$）内达到新的频率。INTOSC 时钟会在 1 ms 内稳定到新的频点。在此变动期间，代码会继续执行。不会有任何迹象表明时钟发生了改变。

OSCTUNE 寄存器还包含了 INTSRC 位。当选择了 31 kHz 频率选项后，用户可通过 INTSRC 位选择用作时钟源的内部振荡器。第 **2.4.1 节 "振荡器控制寄存器"** 中将对此进行详细讨论。

2.2.5.3 内部振荡器输出频率和漂移

出厂时已校准了内部振荡器电路使之能够产生 8.0 MHz 的 INTOSC 输出频率。但是，此频率可能会随着 V_{DD} 电压或温度的改变而发生漂移，这可能会影响控制器运行。

低频 INTRC 振荡器的工作独立于 INTOSC 源。电压和温度变化导致的 INTOSC 变化并不一定会使 INTRC 变化，反之亦然。

寄存器 2-1： OSCTUNE：振荡器调节寄存器

R/W-0	U-0	U-0	R/W-0	R/W-0	R/W-0	R/W-0	R/W-0
INTSRC	—	—	TUN4	TUN3	TUN2	TUN1	TUN0
bit 7							bit 0

bit 7 **INTSRC**：内部振荡器低频源选择位
1 = 来自 8 MHz INTOSC 源的 31.25 kHz 器件时钟 （使能 256 分频）
0 = 直接来自 INTRC 内部振荡器的 31 kHz 器件时钟

bit 6-5 未用：读为 0

bit 4-0 **TUN4:TUN0**：频率调节位
01111 = 最高频率
• •
• •
00001
00000 = 中间频率。振荡器模块运行在已校准后的频率上。
11111
• •
• •
10000 = 最低频率

图注：			
R = 可读位	W = 可写位	U = 未用位，读为 0	
-n = 上电复位时的值	1 = 置 1	0 = 清零	x = 未知

2.2.5.4 补偿 INTOSC 漂移

通过修改 OSCTUNE 寄存器的值可以调节 INTOSC 的频率。这对 INTRC 时钟源的频率没有影响。

调节 INTOSC 时钟源需要了解调节的时间、调节的方向以及在某些情况下的改变量。例如使用EUSART时，在它开始产生帧错误，或者在异步模式下接收数据有错误时就需要进行调节。帧错误表示器件时钟的频率太高；要对此进行调节，可以减小 OSTUNE 寄存器中的值以降低时钟频率。另一方面，数据中有错误则表明时钟速率太低；要补偿，可以增大 OSTUNE 寄存器中的值来提高时钟频率。

也可以将器件时钟速率与参考时钟的速率作比较。这时要用到两个定时器；一个定时器由外设时钟提供时钟源，而另一个定时器由一个固定的参考源 （如 Timer1 振荡器）提供时钟源。两个定时器都被清零，但由参考源提供时钟信号的定时器产生中断。当中断发生时，使用内部时钟源的定时器的当前值被读取且两个定时器都被清零。如果使用内部时钟源的定时器的值大于期望值，则表示内部振荡器电路运行过快。要对此进行调节，需减小 OSCTUNE 寄存器中的值。

CCP 模块可以使用独立运行的 Timer1 （或 Timer3），Timer1 （或 Timer3）由内部振荡器电路和已知周期（例如， AC 电源频率）的外部事件提供时钟源。CCPRxH:CCPRxL 寄存器捕捉第一个事件的时间，并进行记录以备后用。当第二个事件引起捕捉时，要从第二个事件的时间中减去第一个事件的时间。由于外部事件的周期是已知的，可以计算两者之间的时间差。

如果测量所得的时间远大于计算所得的时间，说明内部振荡器电路运行过快；要补偿，就要减小 OSCTUNE 寄存器的值。如果测量所得的时间远小于计算所得的时间，说明内部振荡器电路运行过慢；要补偿，就要增大 OSCTUNE 寄存器的值。

2.3 USB 的振荡器设置

当 PIC18F4550 进行 USB 连接时，它必须提供 6 MHz 或 48 MHz 的时钟以进行 USB 操作，时钟的频率取决于使用的是低速模式还是全速模式。在选择振荡器频率和对器件编程前应先做出决定。

表 2-3 中所示为所有与 USB 操作兼容的可用振荡器配置。

2.3.1 低速操作

低速模式的 USB 时钟信号来自于主振荡器，而并非直接来自 PLL。它被四分频以产生实际频率为 6 MHz 的时钟信号。因此，当 USB 模块工作时单片机只能使用 24 MHz 的时钟，而且该时钟源必须处于某种主振荡器模式（XT、 HS 或 EC，带或不带 PLL）。

如果单片机时钟源是辅助振荡器或内部振荡电路，此限制则不适用。

2.3.2 运行不同的 USB 和单片机时钟

USB 模块在两种模式下都能异步于单片机内核和其他外设运行。这意味着可以在使用主振荡器作为 USB 时钟的同时，用一个独立的时钟源以较低的速率驱动单片机。如果必须只使用一个时钟源为所有模块提供时钟，采用全速模式可选择较广范围的单片机时钟频率。

表 **2-3**： **USB 的振荡器配置选项**

输入振荡器频率	PLL 分频（PLLDIV2:PLLDIV0）	时钟模式（FOSC3:FOSC0）	MCU 时钟分频（CPUDIV1:CPUDIV0）	单片机时钟频率
48 MHz	N/A[1]	EC 和 ECIO	不分频（00）	48 MHz
			÷2 （01）	**24 MHz**
			÷3 （10）	16 MHz
			÷4 （11）	12 MHz
48 MHz	÷12（111）	EC 和 ECIO	不分频（00）	48 MHz
			÷2（01）	**24 MHz**
			÷3（10）	16 MHz
			÷4（11）	12 MHz
		ECPLL 和 ECPIO	÷2 （00）	48 MHz
			÷3 （01）	32 MHz
			÷4 （10）	**24 MHz**
			÷6 （11）	16 MHz
40 MHz	÷10 （110）	EC 和 ECIO	不分频（00）	40 MHz
			÷2 （01）	20 MHz
			÷3 （10）	13.33 MHz
			÷4 （11）	10 MHz
		ECPLL 和 ECPIO	÷2 （00）	48 MHz
			÷3 （01）	32 MHz
			÷4 （10）	**24 MHz**
			÷6 （11）	16 MHz
24 MHz	÷6 （101）	HS、 EC 和 ECIO	不分频（00）	**24 MHz**
			÷2 （01）	12 MHz
			÷3 （10）	8 MHz
			÷4 （11）	6 MHz
		HSPLL、 ECPLL 和 ECPIO	÷2 （00）	48 MHz
			÷3 （01）	32 MHz
			÷4 （10）	**24 MHz**
			÷6 （11）	16 MHz

图注： 除了 24 MHz，其他所有时钟频率都只与全速 USB 操作（USB 时钟为 48 MHz）相关。**粗体**用来突出兼容低速 USB 操作的时钟频率（系统时钟为 24 MHz， USB 时钟为 6 MHz）。

注 **1：** 仅当 USBDIV 配置位清零时有效。

表 **2-3**： **USB** 的振荡器配置选项 （续）

输入振荡器频率	**PLL** 分频 （**PLLDIV2:PLLDIV0**）	时钟模式 （**FOSC3:FOSC0**）	**MCU** 时钟分频 （**CPUDIV1:CPUDIV0**）	单片机 时钟频率
20 MHz	÷5 （100）	HS、 EC 和 ECIO	不分频 （00）	20 MHz
			÷2 （01）	10 MHz
			÷3 （10）	6.67 MHz
			÷4 （11）	5 MHz
		HSPLL、 ECPLL 和 ECPIO	÷2 （00）	48 MHz
			÷3 （01）	32 MHz
			÷4 （10）	**24 MHz**
			÷6 （11）	16 MHz
16 MHz	÷4 （011）	HS、 EC 和 ECIO	不分频 （00）	16 MHz
			÷2 （01）	8 MHz
			÷3 （10）	5.33 MHz
			÷4 （11）	4 MHz
		HSPLL、 ECPLL 和 ECPIO	÷2 （00）	48 MHz
			÷3 （01）	32 MHz
			÷4 （10）	**24 MHz**
			÷6 （11）	16 MHz
12 MHz	÷3 （010）	HS、 EC 和 ECIO	不分频 （00）	12 MHz
			÷2 （01）	6 MHz
			÷3 （10）	4 MHz
			÷4 （11）	3 MHz
		HSPLL、 ECPLL 和 ECPIO	÷2 （00）	48 MHz
			÷3 （01）	32 MHz
			÷4 （10）	**24 MHz**
			÷6 （11）	16 MHz
8 MHz	÷2 （001）	HS、 EC 和 ECIO	不分频 （00）	8 MHz
			÷2 （01）	4 MHz
			÷3 （10）	2.67 MHz
			÷4 （11）	2 MHz
		HSPLL、 ECPLL 和 ECPIO	÷2 （00）	48 MHz
			÷3 （01）	32 MHz
			÷4 （10）	**24 MHz**
			÷6 （11）	16 MHz
4 MHz	÷1 （000）	XT、 HS、 EC 和 ECIO	不分频 （00）	4 MHz
			÷2 （01）	2 MHz
			÷3 （10）	1.33 MHz
			÷4 （11）	1 MHz
		HSPLL、 ECPLL、 XTPLL 和 ECPIO	÷2 （00）	48 MHz
			÷3 （01）	32 MHz
			÷4 （10）	**24 MHz**
			÷6 （11）	16 MHz

图注： 除了 24 MHz，其他所有时钟频率都只与全速 USB 操作 （USB 时钟为 48 MHz）相关。**粗体**用来突出兼容低速 USB 操作的时钟频率 （系统时钟为 24 MHz， USB 时钟为 6 MHz）。

注 1： 仅当 USBDIV 配置位清零时有效。

2.4 时钟源与振荡器切换

与以前的 PIC18 增强型器件一样， PIC18F2455/2550/4455/4550 系列允许将器件时钟源从主振荡器切换到备用低频时钟源。PIC18F2455/2550/4455/4550 器件提供了两个备用时钟源。当使能了备用时钟源时，各种功耗管理工作模式都可使用。

基本上，这些器件有三种时钟源：

- 主振荡器
- 辅助振荡器
- 内部振荡器电路

主振荡器包括外部晶振和谐振器模式、外部时钟模式和内部振荡器电路。具体使用哪一种模式由 FOSC3:FOSC0 配置位定义。这些模式已在本章前面的内容中作过详细介绍。

辅助振荡器是不与 OSC1 或 OSC2 引脚连接的外部时钟源。这些时钟源即使在控制器处于功耗管理模式时仍然可以继续工作。

PIC18F2455/2550/4455/4550 器件提供 Timer1 振荡器作为辅助振荡器。该振荡器（在所有的功耗管理模式下）通常是实时时钟等功能的时基。大部分情况下，在 RC0/T1OSO/T13CKI 和 RC1/T1OSI/UOE 引脚之间接有一个 32.768 kHz 的时钟晶振。与 XT 和 HS 模式振荡电路相同，每个引脚与地之间也都接有负载电容。在**第 12.3 节 “Timer1 振荡器”**中对 Timer1 振荡器作了更详细的讨论。

除了作为主时钟源外，**内部振荡器电路**还可以作为功耗管理模式时钟源。INTRC 源也可以作为几种特殊功能的时钟源，例如 WDT 和故障保护时钟监视器。

2.4.1 振荡器控制寄存器

OSCCON 寄存器（寄存器 2-2）控制全功耗模式和功耗管理模式下器件时钟工作的多个方面。

系统时钟选择位 SCS1:SCS0 用于选择时钟源。可用的时钟源包括主时钟（由 FOSC3:FOSC0 配置位定义）、辅助时钟（Timer1 振荡器）和内部振荡器电路。在写入一个或多个位后，有一段短的时钟切换间隔，之后，时钟源发生改变。在所有形式的复位后 SCS 位都会被清零。

内部振荡器频率选择位 IRCF2:IRCF0，用于选择驱动器件时钟的内部振荡器电路的输出频率。这些频率可以是 INTRC 源的频率、INTOSC 源的频率（8 MHz）或 INTOSC 后分频器产生的几个频率之一（31 kHz 到 4 MHz）。如果器件时钟源由内部振荡器电路提供，改变这些位的状态会使内部振荡器输出立即发生改变。当器件复位时，内部振荡器电路的默认输出频率设为 1 MHz。

当选定了 31 kHz 的输出频率（IRCF2:IRCF0 = 000）时，用户可以选择哪个内部振荡器充当时钟源。这通过设置 OSCTUNE 寄存器中的 INTSRC 位（OSCTUNE<7>）完成。通过使能 INTOSC 后分频器的 256 分频输出，将该位置 1 选择 INTOSC 作为 31.25 kHz 的时钟源。清零 INTSRC 位选择 INTRC（标称值为 31 kHz）作为时钟源。

此选项使用户能选择可调节且更精确的 INTOSC 作为时钟源，同时以非常低的时钟速率运行以节约功耗。无论 INTSRC 的设置如何，INTRC 总是作为看门狗定时器和故障保护时钟监视器之类部件的时钟源。

OSTS、IOFS 和 T1RUN 位表明当前提供器件时钟的时钟源。 OSTS 位置 1 表明振荡器起振定时器延时已结束且主时钟在主时钟模式下提供器件时钟源。IOFS 位置 1 表明内部振荡器电路已稳定并在 RC 时钟模式下提供器件时钟源。 T1RUN 位（T1CON<6>）置 1 表明 Timer1 振荡器正在辅助时钟模式下提供器件时钟源。在功耗管理模式下，任何时候这三个位中只有一个会被置 1。如果这些位都没有置 1，则表示当前时钟源是 INTRC，或内部振荡器电路刚刚起振且尚未稳定。

IDLEN 位决定当执行 SLEEP 指令时，器件是否进入休眠模式或者某种空闲模式。

第 3.0 节 “功耗管理模式”中详细讨论了 OSCCON 寄存器中的标志位和控制位的使用。

> 注 **1**：要选择辅助时钟源，必须使能 Timer1 振荡器。通过将 Timer1 控制寄存器中的 T1OSCEN 位（T1CON<3>）置 1，可以使能 Timer1 振荡器。如果未使能 Timer1 振荡器，选择辅助时钟源的任何尝试都会被忽略。
>
> **2**：建议当Timer1振荡器稳定工作之后再将它切换为时钟源；否则当Timer1振荡器起振时可能会发生很长的延时。

2.4.2 振荡器转换

PIC18F2455/2550/4455/4550 器件包含了防止在切换时钟源时发生时钟“毛刺”的电路。在时钟切换时，系统时钟会有短暂停顿。停顿的长度是旧时钟源的两个周期加上新时钟源的三到四个周期的和。此公式假设新时钟源是稳定的。

第 3.1.2 节“进入功耗管理模式”中对时钟转换进行了更详细的讨论。

寄存器 2-2： OSCCON：振荡器控制寄存器

R/W-0	R/W-1	R/W-0	R/W-0	R[(1)]	R-0	R/W-0	R/W-0
IDLEN	IRCF2	IRCF1	IRCF0	OSTS	IOFS	SCS1	SCS0
bit 7							bit 0

bit 7 **IDLEN：**空闲使能位
1 = 在执行 SLEEP 指令后器件进入空闲模式
0 = 在执行 SLEEP 指令后器件进入休眠模式

bit 6-4 **IRCF2:IRCF0：**内部振荡器频率选择位
111 = 8 MHz （INTOSC 直接驱动时钟）
110 = 4 MHz
101 = 2 MHz
100 = 1 MHz[(3)]
011 = 500 kHz
010 = 250 kHz
001 = 125 kHz
000 = 31 kHz （来自 INTOSC/256 或直接来自 INTRC）[(2)]

bit 3 **OSTS：**振荡器起振延时状态位 [(1)]
1 = 振荡器起振定时器延时已经结束；主振荡器正在运行
0 = 振荡器起振定时器正在进行延时；主振荡器尚未准备就绪

bit 2 **IOFS：**INTOSC 频率稳定位
1 = INTOSC 频率已稳定
0 = INTOSC 频率尚未稳定

bit 1-0 **SCS1:SCS0：**系统时钟选择位
1x = 内部振荡器电路
01 = Timer1 振荡器
00 = 主振荡器

注 **1：**取决于 IESO 配置位的状态。
2：时钟源由 INTSRC 位 （OSCTUNE<7>）选择，参见本文。
3：复位时 INTOSC 的默认输出频率。

图注：			
R = 可读位	W = 可写位	U = 未用位，读为 0	
-n = 上电复位时的值	1 = 置 1	0 = 清零	x = 未知

 初稿

2.5 功耗管理模式对各种时钟源的影响

当选择了 PRI_IDLE 模式时，指定的主振荡器会继续运行而不中断。对于所有其他功耗管理模式，使用 OSC1 引脚的振荡器会被禁止。除非使能 USB 模块，否则 OSC1 引脚将停止振荡。如果振荡器使用 OSC2 引脚，则该引脚也将停止振荡。

在辅助时钟模式下（SEC_RUN和SEC_IDLE），Timer1 振荡器作为器件时钟源工作。如果需要，Timer1 振荡器也可以运行在所有功耗管理模式下为 Timer1 或 Timer3 提供时钟源。

在内部振荡器模式（RC_RUN 和 RC_IDLE）下，由内部振荡器电路提供器件时钟源。无论处于何种功耗管理模式下，都可以直接使用 31 kHz 的 INTRC 输出来提供时钟源或者使能它来支持多种特殊功能（欲知更多有关 WDT、故障保护时钟监视器和双速启动的信息，请参见**第 25.2 节“看门狗定时器（WDT）”**、**第 25.3 节“双速启动”**和**第 25.4 节“故障保护时钟监视器”**）。8 MHz 的 INTOSC 输出可以直接用于为器件提供时钟源，或者先由后分频器进行分频。如果直接将 INTRC 输出作为时钟源，则会禁止 INTOSC 输出。

无论选择了何种运行或空闲模式，USB 时钟将继续工作。如果将晶振或基于谐振器的振荡器作为器件时钟源，该振荡器将继续为 USB 模块提供时钟信号；内核和所有其他模块将切换到新的时钟源。

如果选择了休眠模式，所有时钟源都会停止。因为休眠模式消除了所有的晶体管开关电流，所以该模式能实现最小的器件电流消耗（仅泄漏电流）。

当 USB 模块在工作和处于已连接状态时，不要启动休眠模式。惟一例外的情况是当器件通过 USB 收到“Suspend”（暂停）命令时。当该模块暂停工作并转变到低功耗状态之后，单片机可以安全地进入休眠模式。

在休眠期间使能任何片上功能都将会增加休眠时的电流消耗。需要使能 INTRC 来支持 WDT 工作。Timer1 振荡器可以用来为实时时钟提供时钟源。不需要器件时钟源的其他功能也可以工作（即，SSP 从器件、PSP 以及 INTn 引脚等）。**第 28.2 节“DC 特性：掉电和供电电流”**列出了会明显增加电流消耗的外设。

2.6 上电延时

有两个定时器用于控制上电延时，这样大部分应用都无需外接复位电路。上电延时可以确保在器件电源稳定（常规环境下）和主时钟稳定工作之前器件保持复位状态。欲知有关上电延时的其他信息，请参见**第 4.5 节“器件复位定时器”**。

第一个定时器是上电延时定时器（Power-up Timer，PWRT），在上电时它提供了固定的延时（表 28-12 中参数33）。通过将PWRTEN配置位清零（= 0）使能它。

第二个定时器是振荡器起振定时器（Oscillator Start-up Timer，OST），用于在晶振稳定前使单片机保持在复位状态（XT 和 HS 模式）。OST 通过计数 1024 个振荡周期实现此延时并在延时后允许振荡器为器件提供时钟。

当选定 HSPLL 振荡器模式时，在 HS 模式下的 OST 延时之后，器件还会在复位状态另外保持 2 ms，这样 PLL 可以锁定到输入时钟频率。

在上电复位之后，会有一段延时间隔 T_{CSD}（表 28-12 中参数 38），控制器在这段时间中为执行指令做准备。此延时与其他延时同时进行。当 EC 或内部振荡器模式用作主时钟源时，这可能是惟一的延时。

表 **2-4：** 休眠模式下 **OSC1** 和 **OSC2** 引脚状态

振荡器模式	**OSC1** 引脚	**OSC2** 引脚
INTCKO	悬空，由外部时钟驱动	处于逻辑低电平（输出时钟 /4）
INTIO	悬空，由外部时钟驱动	配置为 PORTA 的 bit 6
ECIO 和 ECPIO	悬空，由外部时钟驱动	配置为 PORTA 的 bit 6
EC	悬空，由外部时钟驱动	处于逻辑低电平（输出时钟 /4）
XT 和 HS	处于静态电平时，反馈反相器被禁用	处于静态电平时，反馈反相器被禁用

注： 欲知由于休眠和 $\overline{MCLR}$ 复位而引起的延时的信息，请参见**第 4.0 节“复位”**中的表 4-2。

注：

3.0 功耗管理模式

PIC18F2455/2550/4455/4550 器件总共提供七种工作模式，可以更有效的进行功耗管理。这些工作模式提供了多种选项，便于在资源受限的应用（即电池供电的设备）中节省功耗。

功耗管理模式分为以下三类：

- 运行模式
- 空闲模式
- 休眠模式

这些分类的依据是为器件的哪些部分提供时钟，有时还包括时钟速率。运行和空闲模式可以使用三种可用时钟源（主时钟源、辅助时钟源或内部振荡器电路）中的任意一种；而休眠模式则不使用时钟源。

功耗管理模式包括几种在以前的 PICmicro® 器件上提供的节约功耗功能。其中之一就是在其他的 PIC18 器件上提供的时钟切换功能，允许控制器使用 Timer1 振荡器代替主振荡器。节省功耗的功能还包括所有 PICmicro 器件都提供的休眠模式，在此模式下所有的器件时钟均停止。

3.1 选择功耗管理模式

选择功耗管理模式之前要决定是否为 CPU 提供时钟以及选择何种时钟源。IDLEN 位（OSCCON<7>）控制是否为 CPU 提供时钟，而 SCS1:SCS0 位（OSCCON<1:0>）用于选择时钟源。表 3-1 总结了各个模式下的位设置、时钟源和受影响的模块。

3.1.1 时钟源

SCS1:SCS0 位可以为功耗管理模式选择三个时钟源中的一个。它们是：

- 主时钟，由 FOSC3:FOSC0 配置位定义
- 辅助时钟 （Timer1 振荡器）
- 内部振荡器电路 （用于 RC 模式）

3.1.2 进入功耗管理模式

通过装载 OSCCON 寄存器可以从一种功耗管理模式切换到另一种功耗管理模式。 SCS1:SCS0 位选择时钟源并决定使用运行模式还是空闲模式。更改这些位将导致立即切换到新的时钟源 （假定新的时钟源正在运行）。切换也可能会带来时钟转换延迟。**第 3.1.3 节 "时钟转换和状态指示位"** 和其后的章节将会讨论这些问题。

执行 SLEEP 指令可以触发进入功耗管理空闲模式或休眠模式。具体进入哪个模式由 IDLEN 位的状态决定。

更改功耗管理模式并不总是需要设置所有这些位，这由当前模式和将要切换的模式决定。多数情况下，可以通过在发出 SLEEP 指令之前更改振荡器选择位或更改 IDLEN 位来进行模式的转换。如果已经正确地配置了 IDLEN 位，则只需执行 SLEEP 指令就能切换到所需的模式。

表 3-1： 功耗管理模式

模式	OSCCON 位		模块时钟		可用时钟和振荡器源
	IDLEN[1]	SCS1:SCS0	CPU	外设	
休眠模式	0	N/A	不工作	关闭	无——所有时钟被禁止
PRI_RUN	N/A	00	为其提供时钟信号	为其提供时钟信号	主时钟——所有振荡器模式。这是正常的全功耗执行模式。
SEC_RUN	N/A	01	为其提供时钟信号	为其提供时钟信号	辅助时钟——Timer1 振荡器
RC_RUN	N/A	1x	为其提供时钟信号	为其提供时钟信号	内部振荡器电路 [2]
PRI_IDLE	1	00	不工作	为其提供时钟信号	主时钟——所有振荡器模式
SEC_IDLE	1	01	不工作	为其提供时钟信号	辅助时钟——Timer1 振荡器
RC_IDLE	1	1x	不工作	为其提供时钟信号	内部振荡器电路 [2]

注 1： 仅当执行 SLEEP 指令时， IDLEN 才反映出有效值。
2： 包括 INTOSC 和 INTOSC 后分频器以及 INTRC 源。

3.1.3 时钟转换和状态指示位

在两个时钟源之间进行切换的时间长度是旧时钟源的 2 个周期与新时钟源的 3 到 4 个周期的和。此公式假设新时钟源是稳定的。

以下 3 位用于表明当前的时钟源及其状态。它们是：

- OSTS （OSCCON<3>）
- IOFS （OSCCON<2>）
- T1RUN （T1CON<6>）

一般来说，在一个给定的功耗管理模式下，这 3 位中只有 1 位会置 1。当 OSTS 位置 1 时，表明由主时钟提供器件时钟。当 IOFS 位置 1 时，表明由 INTOSC 提供稳定的 8MHz 时钟给分频器，由分频器驱动器件时钟。当 T1RUN 位置 1 时，表明由 Timer1 振荡器提供时钟。如果这些位均不置 1，则要么是由 INTRC 时钟源为器件提供时钟信号，要么此时 INTOSC 源尚未稳定。

如果用FOSC3:FOSC0配置位将内部振荡器电路配置为主时钟源，则在 PRI_RUN 或 PRI_IDLE 模式中，OSTS 和 IOFS 位可能同时置 1。这表示主时钟 （INTOSC 输出）正在产生稳定的 8 MHz 输出信号。进入工作频率相同的 RC 功耗管理模式将清零 OSTS 位。

> **注 1：** 在仅修改IRCF位时应特别小心。如果VDD 电压小于 3V，要避免选择那些低 VDD 电压不支持的高时钟速率。违反 VDD/FOSC 规范会导致器件运行不正常。
>
> **2：** 执行SLEEP指令不一定会使器件进入休眠模式。该指令充当触发条件，根据 IDLEN 位的设置，使控制器进入休眠模式或某种空闲模式。

3.1.4 多条 SLEEP 命令

进入哪一种功耗管理模式由 SLEEP 指令执行时 IDLEN 位的设置决定。如果又执行了一条 SLEEP 指令，器件将进入那时由IDLEN位指定的功耗管理模式。如果IDLEN 位已更改，则器件将进入由新的设置指定的功耗管理模式。

3.2 运行模式

在运行模式中，内核和外设的时钟均有效。这些模式之间的差异在于时钟源的不同。

3.2.1 PRI_RUN 模式

PRI_RUN 模式是单片机正常工作的全功耗运行模式。如果没有使能双速启动， PRI_RUN 模式也是器件复位后的默认模式（详细信息请参见**第 25.3 节 “双速启动”**）。在此模式下，OSTS 位置 1。如果内部振荡器电路为主时钟源，IOFS 位可能被置 1（见**第 2.4.1 节“振荡器控制寄存器”**）。

3.2.2 SEC_RUN 模式

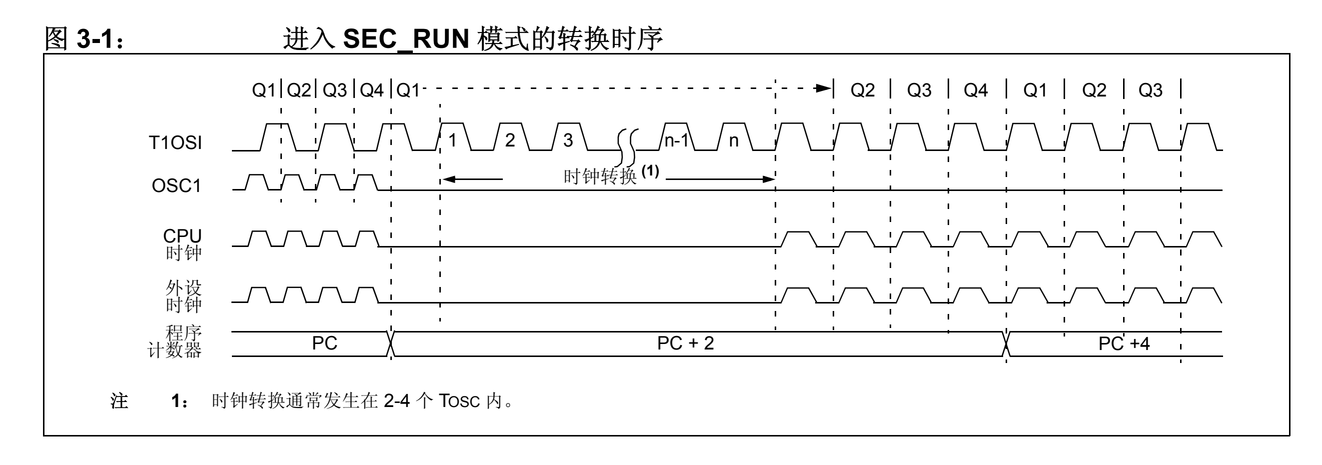
SEC_RUN 模式与其他 PIC18 器件提供的“时钟切换”功能兼容。在此模式下， CPU 和外设由 Timer1 振荡器提供时钟。这让用户在仍使用高精度时钟源的情况下适当地降低功耗。

通过将 SCS1:SCS0 位设置为 `01` 进入 SEC_RUN 模式。器件时钟源切换到 Timer1 振荡器（见图 3-1），关闭主振荡器，T1RUN 位（T1CON<6>）置 1，OSTS 位清零。

> **注：** Timer1 振荡器应该在进入 SEC_RUN 模式之前已经开始运行。如果当SCS1:SCS0位被置 `01` 时， T1OSCEN 位未置 1，就不会进入 SEC_RUN 模式。如果 Timer1 振荡器已经被使能，但没有开始运行，器件时钟将被延时直到振荡器起振为止。在这种情况下，初始振荡器运行很不稳定，可导致无法预料的运行结果。

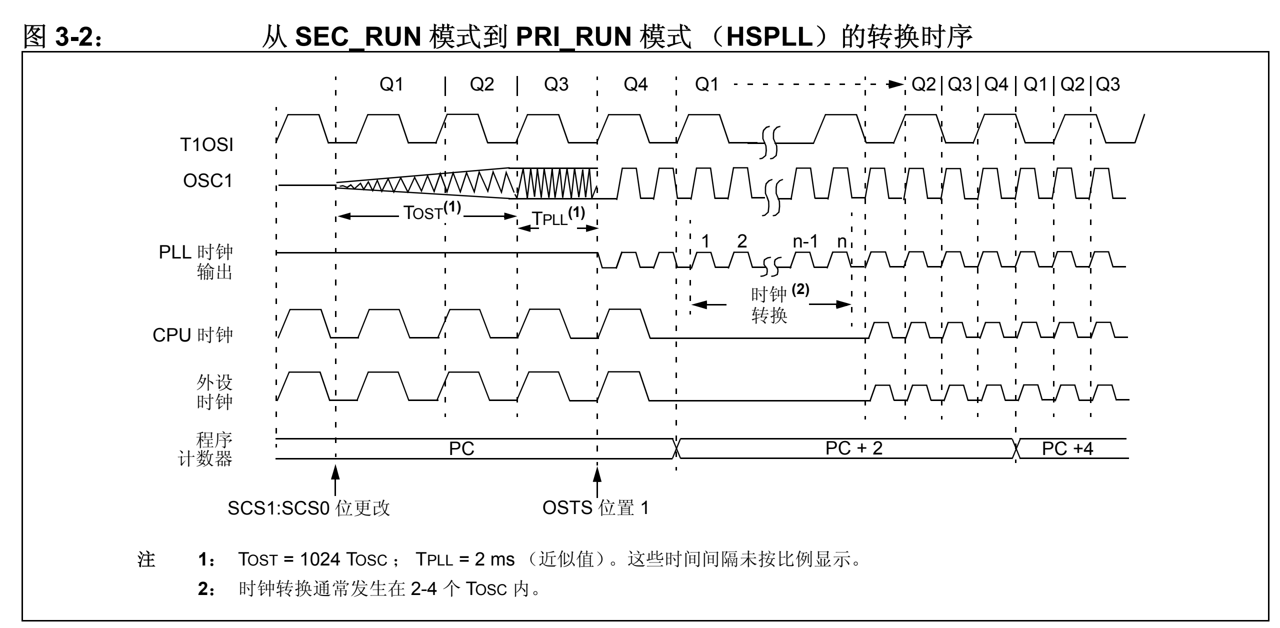
从 SEC_RUN 模式转换到 PRI_RUN 模式时，在主时钟起振期间外设和 CPU 继续将 Timer1 振荡器用作时钟源。当主时钟准备就绪以后，时钟切换回主时钟 （见图 3-2）。当时钟切换完成后，T1RUN位被清零，OSTS 位被置 1 并且由主时钟提供器件时钟。唤醒不会影响 IDLEN 和 SCS 位。 Timer1 振荡器继续运行。

图 3-1：　　进入 SEC_RUN 模式的转换时序



注　1：时钟转换通常发生在 2-4 个 TOSC 内。

图 3-2：　　从 SEC_RUN 模式到 PRI_RUN 模式 （HSPLL）的转换时序



注　1：TOST = 1024 TOSC； TPLL = 2 ms （近似值）。这些时间间隔未按比例显示。
　　2：时钟转换通常发生在 2-4 个 TOSC 内。

3.2.3　　RC_RUN 模式

在 RC_RUN 模式下，使用内部振荡器作为 CPU 和外设的时钟源；主时钟关闭。在使用 INTRC 时钟源时，此模式是所有仍能执行代码的运行模式中最节省功耗的。它非常适用于对时间精度要求不高或者并不一直需要高速时钟的应用。

如果主时钟源为内部振荡器电路 （INTRC 或 INTOSC），在工作期间，PRI_RUN 和 RC_RUN 这两种模式区别不大。但是在进入和退出 RC_RUN 模式时会发生时钟切换延时。因此，如果主时钟源为内部振荡器电路，建议不要使用 RC_RUN 模式。

通过将SCS1位置1可以进入此模式。虽然未使用SCS0位，但还是建议将它清零，从而保持与后续器件的软件兼容性。当时钟源切换到 INTOSC 多路复用器时 （见图 3-3），主振荡器将被关闭并且 OSTS 位被清零。在任何时候更改 IRCF 位，可以立即更改时钟速率。

> **注：**　在仅修改 IRCF 位时应特别小心。如果 VDD 电压小于 3V，要避免选择那些低 VDD 电压不能支持的高时钟速率。违反 VDD/FOSC 规范会导致器件运行不正常。

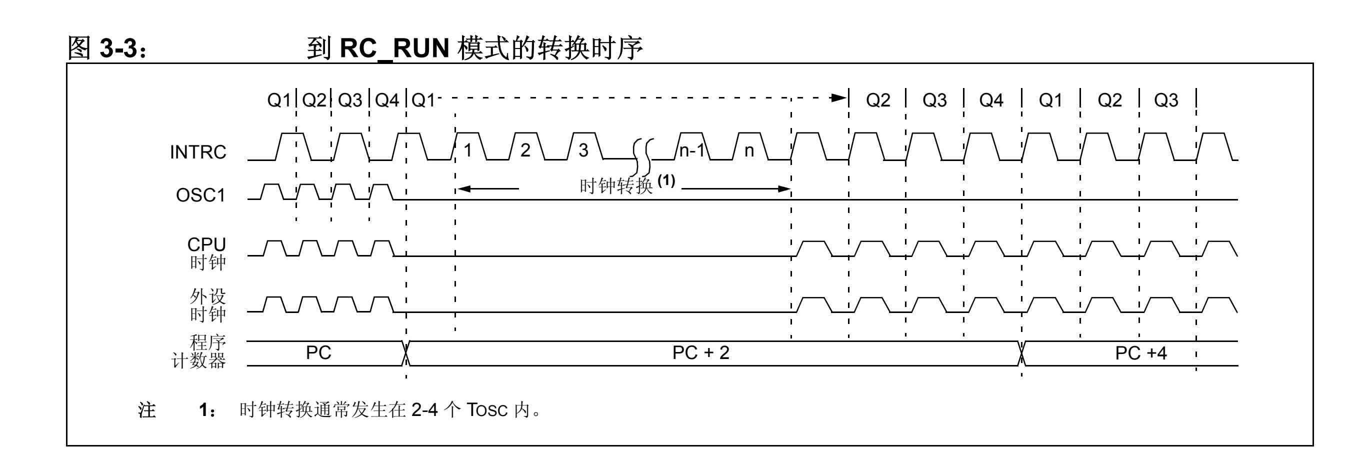
如果 IRCF 位和 INTSRC 位全部被清零，就会禁止 INTOSC 输出且 IOFS 位保持清零。此时不会有当前时钟源的任何指示信息。由 INTRC 时钟源提供器件时钟。

如果 IRCF 位不全为零（因而使能 INTOSC 输出），或者 INTSRC 被置 1， IOFS 位就会在 INTOSC 输出稳定后置 1。在一个 TIOBST 间隔之后，INTOSC 时钟源趋于稳定，此时器件时钟继续运行。

如果 IRCF 位在先前已被设置为一个非零值，或者 INTSRC 在 SCS1 置 1 之前就已经置 1 并且 INTOSC 源已经稳定，那么 IOFS 位将保持置 1 状态。

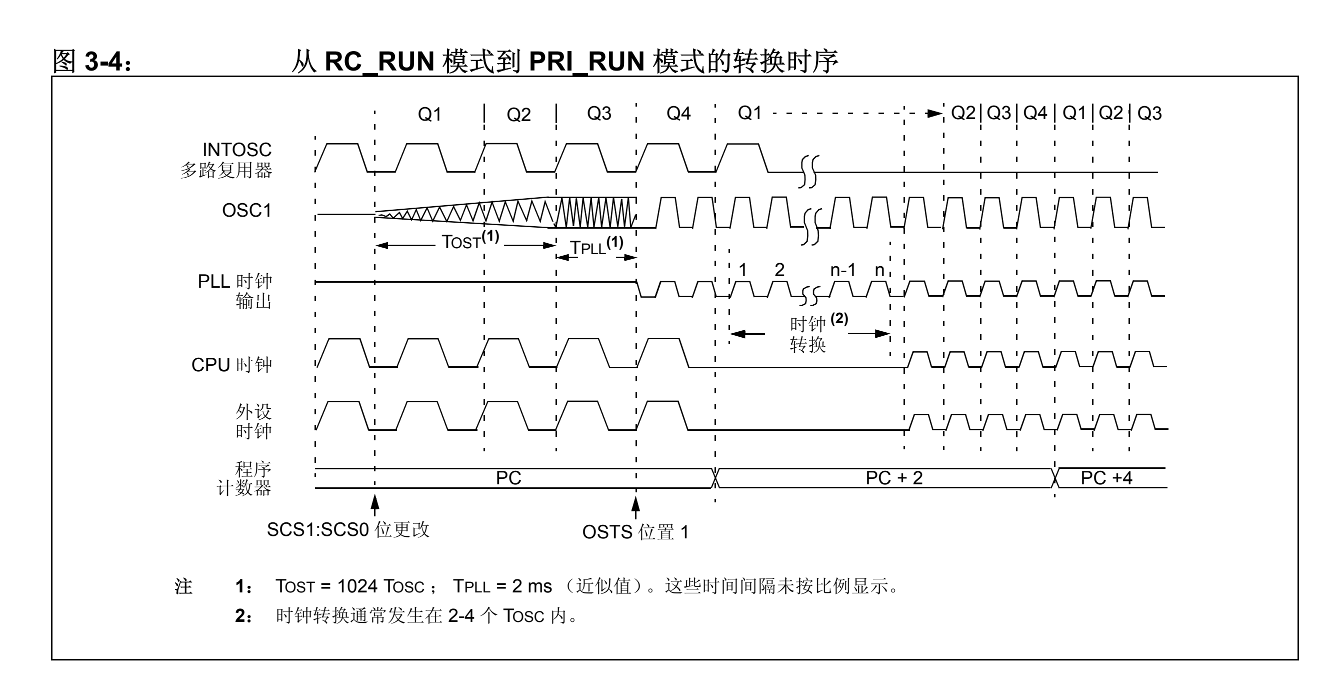
从 RC_RUN 模式转换到 PRI_RUN 模式时，在主时钟起振期间器件将继续使用 INTOSC 多路复用器作为时钟源。当主时钟准备就绪以后，时钟切换到主时钟（见图 3-4）。当时钟切换完成后，IOFS 位被清零，OSTS 位被置 1 并且由主时钟提供器件时钟。这种切换不会影响 IDLEN 和 SCS 位。如果使能了 WDT 或故障保护时钟监视器， INTRC 源将继续运行。

图 3-3： 到 RC_RUN 模式的转换时序



注 1： 时钟转换通常发生在 2-4 个 TOSC 内。

图 3-4： 从 RC_RUN 模式到 PRI_RUN 模式的转换时序



注 1： TOST = 1024 TOSC ； TPLL = 2 ms （近似值）。这些时间间隔未按比例显示。

2： 时钟转换通常发生在 2-4 个 TOSC 内。

 初稿

3.3 休眠模式

PIC18F2455/2550/4455/4550 器件的功耗管理休眠模式和所有其他PICmicro器件提供的传统休眠模式相同。通过清零 IDLEN 位 （器件复位时的默认状态）并执行 SLEEP 指令即可进入此模式。这将关闭选定的振荡器（图 3-5）并清零所有的时钟源状态位。

从其他模式进入休眠模式不需要时钟切换。这是因为单片机一旦进入休眠模式就不需要时钟了。如果选择了 WDT，INTRC 源将继续运行。如果使能了 Timer1 振荡器， Timer1 也将继续运行。

在休眠模式中发生唤醒事件时（由于中断、复位或 WDT 超时），在由 SCS1:SCS0 位选定的时钟源准备就绪之前，器件将没有时钟源 （见图 3-6），但是如果使能了双速启动或故障保护时钟监视器，它将使用内部振荡器电路作为时钟源（见第 **25.0** 节 **"CPU 的特殊性能"**）。在这两种情况下，当主时钟提供器件时钟时， OSTS 位被置 1。这种唤醒不会影响 IDLEN 和 SCS 位。

3.4 空闲模式

空闲模式允许在外设继续工作的时候有选择地关闭单片机的CPU。选择某种特定的空闲模式使用户能进一步管理功耗。

如果在执行 SLEEP 指令时，IDLEN 位被置为 1，外设将使用由 SCS1:SCS0 位选择的时钟作为时钟源，而 CPU 将没有时钟源。时钟源状态位不受影响。将 IDLEN 置 1 并执行 SLEEP 指令将快速从一种给定的运行模式切换到相应的空闲模式。

如果选择了 WDT， INTRC 时钟源将继续运行。如果使能了 Timer1 振荡器， Timer1 也将继续运行。

由于 CPU 没有执行指令，器件只能通过中断、WDT 超时或复位从任一空闲模式退出。当发生唤醒事件时，CPU 运行将被延迟一个 TCSD 间隔 （表 28-12 中的参数 38），在此延迟期间 CPU 将为执行代码做好准备。当 CPU 开始执行代码时，它将沿用与当前空闲模式相同的时钟源。例如，当从 RC_IDLE 模式唤醒时，将使用内部振荡器电路作为 CPU 和外设的时钟源 （即 RC_RUN 模式）。这种唤醒不会影响 IDLEN 和 SCS 位。

当处于任何空闲模式或休眠模式时， WDT 超时将导致器件被唤醒，并进入由SCS1:SCS0位指定的运行模式。

图 3-5： 进入休眠模式的转换时序

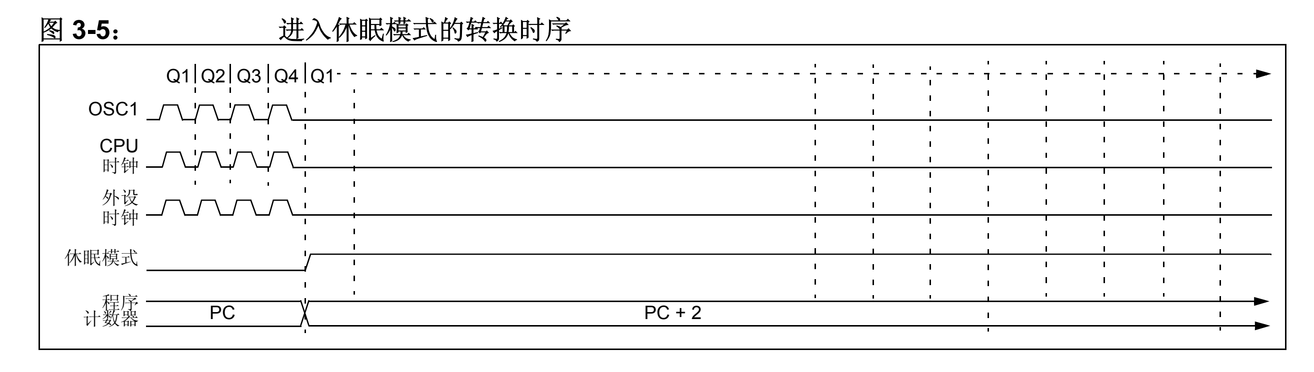
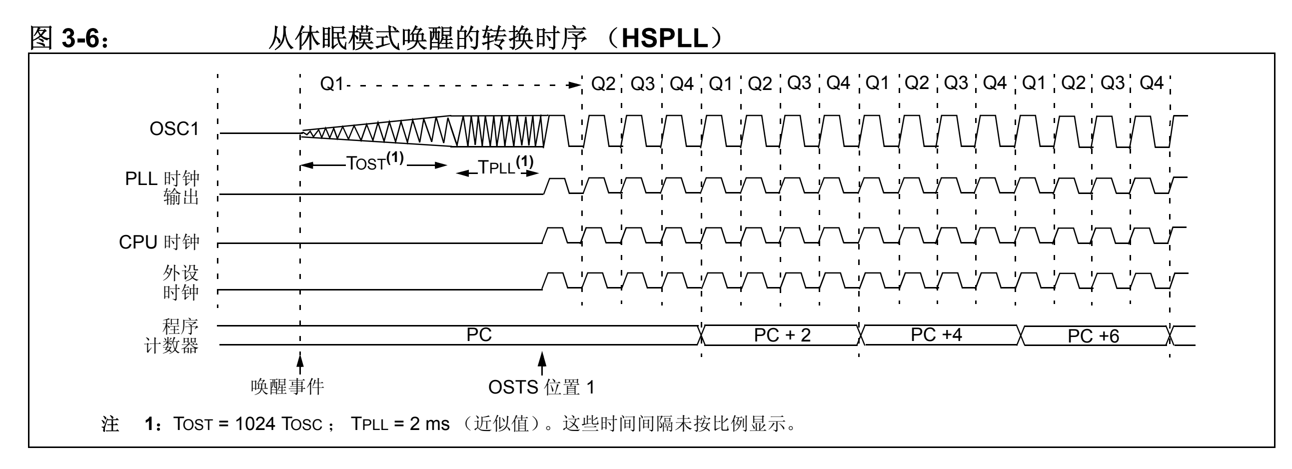


图 3-6： 从休眠模式唤醒的转换时序 （HSPLL）



3.4.1 PRI_IDLE 模式

此模式是三种低功耗空闲模式中惟一不会禁止主器件时钟的空闲模式。对于那些对时序精度要求较高的应用来说，由于时钟源不需要“热身”或从其他振荡器切换过来，选用此模式可以有更加精确的主时钟源，并能够以最快的速度恢复器件运行。

通过将 IDLEN 位置 1 并执行 SLEEP 指令，可以从 PRI_RUN 模式进入 PRI_IDLE 模式。如果器件处于另一种运行模式，可以先将 IDLEN 位置 1，然后将 SCS 位清零并执行 SLEEP。虽然 CPU 已被禁止，但外设仍可继续使用由FOSC3:FOSC0配置位指定的主时钟源。OSTS 位保持置 1 （见图 3-7）。

当发生唤醒事件时，由主时钟源为 CPU 提供时钟。在唤醒事件和代码开始执行之间需要一个 TCSD 的延时。需要这个延时来让 CPU 做好执行指令的准备。在唤醒之后，OSTS 位保持置 1。唤醒不会影响 IDLEN 和 SCS 位（见图 3-8）。

3.4.2 SEC_IDLE 模式

在 SEC_IDLE 模式中， CPU 被禁止，但外设继续将 Timer1 振荡器作为时钟源。通过置位 IDLEN 位并执行 SLEEP 指令，可以从 SEC_RUN 进入此模式。如果器件处于其他运行模式，先将 IDLEN 置 1，然后将 SCS1:SCS0 设置为 01 并执行 SLEEP。当时钟源切换到 Timer1 振荡器时，主振荡器关闭， OSTS 位清零，T1RUN 位置 1。

当唤醒事件发生时，外设继续将 Timer 振荡器作为时钟源。唤醒事件后经过一个 TCSD 时间间隔，CPU 开始执行代码并使用 Timer1 振荡器作为其时钟源。唤醒不会影响 IDLEN 和 SCS 位。 Timer1 振荡器继续运行 （见图 3-8）。

注：	Timer1 振荡器应该在进入 SEC_IDLE 模式之前就已经运行了。如果执行 SLEEP 指令时T1OSCEN位没有置1，就会忽略SLEEP 指令并且不会进入 SEC_IDLE 模式。如果使能了 Timer1 振荡器，但它尚未运行，外设时钟将会延时直到该振荡器起振。在这种情况下，振荡器一开始运行很不稳定，可能导致无法预料的运行结果。

图 3-7： 进入空闲模式的转换时序

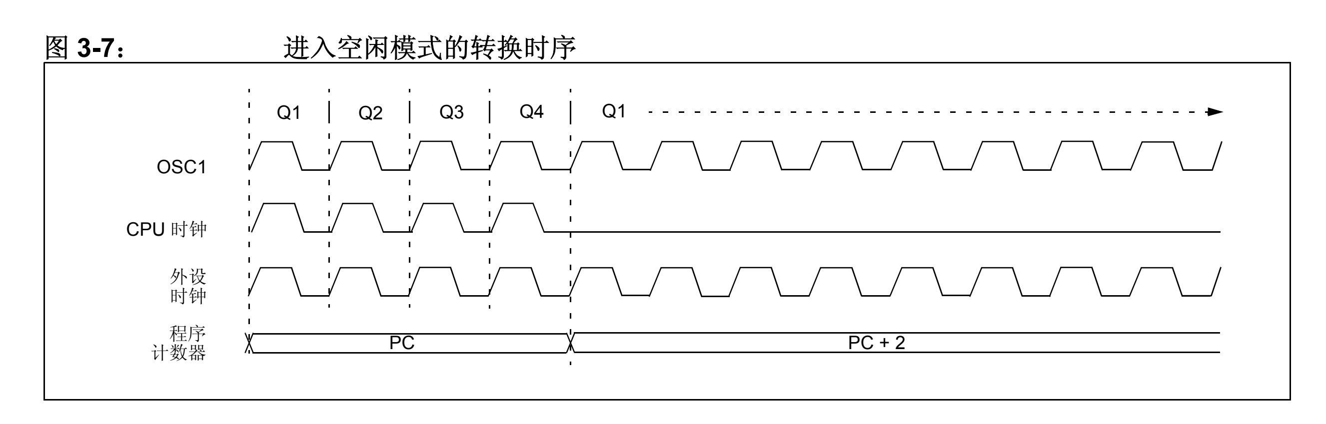
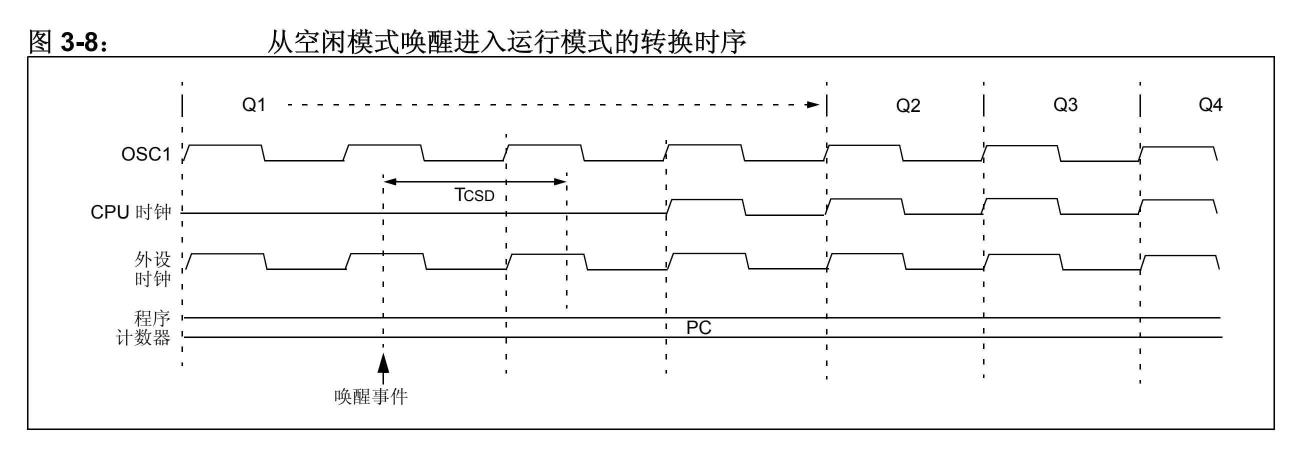


图 3-8： 从空闲模式唤醒进入运行模式的转换时序



 初稿

3.4.3 RC_IDLE 模式

在 RC_IDLE 模式下，CPU 被禁止，但外设仍继续使用内部振荡器电路驱动的 INTOSC 多路复用器作为时钟源。该模式允许在空闲期间对功耗进行控制。

通过将IDLEN位置1并执行SLEEP指令可以从RC_RUN模式进入此模式。如果器件处于另一种运行模式，可以先将IDLEN位置1，然后将SCS1位置1并执行SLEEP。虽然 SCS0 的值将被忽略，但仍建议将其清零，这可保持软件与今后器件兼容。通过在执行 SLEEP 指令之前修改 IRCF 位，可以使用 INTOSC 多路复用器来选择更高的时钟频率。当时钟源切换到 INTOSC 多路复用器时，主振荡器被关闭， OSTS 位被请零。

如果IRCF位被设置为非零值，或者INTSRC位被置1，就会使能INTOSC输出。在一个 TIOBST 间隔（表 28-12 中的参数 39）之后， INTOSC 输出趋于稳定，然后将 IOFS 位置 1。外设的时钟继续运行直到 INTOSC 时钟源趋于稳定。如果 IRCF 位被设置为一个非零值，或者 INTSRC 在执行 SLEEP 之前就已经置 1 并且 INTOSC 源已经稳定，那么 IOFS 位将保持置 1。如果 IRCF 和 INTSRC 位全部清零，就不会使能 INTOSC 输出，IOFS 位将保持清零状态，此时将不会有任何标志位表明当前使用的是哪一个时钟源。

当唤醒事件发生时，外设继续将 INTOSC 多路复用器作为时钟源。唤醒事件发生后经过一个 TCSD 时间间隔，CPU 开始执行代码并使用 INTOSC 多路复用器作为其时钟源。这种唤醒不会影响 IDLEN 和 SCS 位。如果使能了 WDT 或故障保护时钟监视器，INTRC 源将继续运行。

3.5 退出空闲和休眠模式

由中断、复位或 WDT 超时触发退出休眠模式或任何空闲模式。本节将讨论引起退出功耗管理模式的触发条件。在每种功耗管理模式中我们已经讨论了时钟源子系统的工作方式（见**第 3.2 节 "运行模式"**、**第 3.3 节 "休眠模式"**和**第 3.4 节 "空闲模式"**）。

3.5.1 由于中断退出

任何可用的中断源都可以引起器件从空闲模式或休眠模式退出，并进入运行模式。要使能此功能，必须通过在某个 INTCON 或 PIE 寄存器中将中断源的允许位置 1 来允许中断源。当相应的中断标志位置 1 时，触发退出操作。

当通过中断从空闲或休眠模式退出时，如果 GIE/GIEH 位（INTCON<7>）置 1，代码就会跳转到中断矢量处执行。否则代码执行就会继续或恢复，而不会跳转（见**第 9.0 节 "中断"**）。

唤醒事件之后需要一个固定的 TCSD 间隔的延时，器件才会退出休眠和空闲模式。CPU 需要此延时来为执行代码做准备。在此延迟后的第一个时钟周期恢复指令执行。

3.5.2 由于 WDT 超时退出

根据 WDT 超时发生时器件所处的功耗管理模式会引发不同的操作。

如果器件没有执行代码（所有空闲模式和休眠模式），超时将导致从功耗管理模式退出（见**第 3.2 节"运行模式"**和**第 3.3 节 "休眠模式"**）。如果器件正在执行代码（所有运行模式），超时将导致WDT复位（见**第 25.2 节 "看门狗定时器 （WDT）"**）。

执行 SLEEP 或 CLRWDT 指令，或是当前选择的时钟源失效（如果使能了故障保护监视器），或是修改 OSCCON 寄存器中的 IRCF 位（如果器件时钟源为内部振荡器电路），这些均会清零 WDT 定时器和后分频器。

3.5.3 由于复位退出

通常，器件通过振荡器起振定时器（OST）保持在复位状态，直到主时钟就绪。主时钟就绪后，OSTS 位置 1，器件开始执行代码。如果将内部振荡器电路作为新的时钟源，则 IOFS 位将置 1。

从复位到开始执行代码期间的延时由唤醒前后的时钟源以及主时钟振荡器的类型（如果新的时钟源为主时钟）决定。表 3-2 汇总了退出延时。

可以在主时钟就绪之前开始执行代码。如果使能了双速启动（见**第 25.3 节 "双速启动"**）或故障保护时钟监视器（见**第 25.4 节"故障保护时钟监视器"**），器件在清零复位源后马上就开始执行代码。由内部振荡电路驱动的 INTOSC 多路复用器作为代码执行的时钟源。执行代码时，由内部振荡器电路提供时钟源直到主时钟就绪，或者在主时钟就绪前进入功耗管理模式，随后关闭主时钟。

3.5.4 无需振荡器起振延时的退出

某些情况下，从功耗管理模式退出不会引起 OST 延时。有两种情形：

- 主时钟源一直工作的 PRI_IDLE 模式
- 主时钟源不是 XT 或 HS 模式

在这些情况下，主时钟源不需要振荡器起振延时，因为它已经在运行（PRI_IDLE），或者它本来就不需要振荡器起振延时 （EC 和任何内部振荡器模式）。然而，当器件退出休眠和空闲模式时，在唤醒事件后仍需要一个固定的延时 TCSD，以便让 CPU 为执行代码做好准备。在此延迟后的第一个时钟周期恢复指令执行。

表 3-2： 通过复位从休眠模式或任何空闲模式唤醒的退出延时 （按时钟源分类）

单片机时钟源		退出延时	时钟准备状态位 （**OSCCON**）
唤醒之前的时钟源	唤醒之后的时钟源		
主器件时钟 （PRI_IDLE 模式）	XT 和 HS	无	OSTS
	XTPLL 和 HSPLL		
	EC		
	INTOSC[3]		IOFS
T1OSC 或 INTRC[1]	XT 和 HS	TOST[4]	OSTS
	XTPLL 和 HSPLL	TOST + t_{rc}[4]	
	EC	TCSD[2]	
	INTOSC[2]	TIOBST[5]	IOFS
INTOSC[3]	XT 和 HS	TOST[5]	OSTS
	XTPLL 和 HSPLL	TOST + t_{rc}[4]	
	EC	TCSD[2]	
	INTOSC[2]	无	IOFS
无 （休眠模式）	XT 和 HS	TOST[4]	OSTS
	XTPLL 和 HSPLL	TOST + t_{rc}[4]	
	EC	TCSD[2]	
	INTOSC[2]	TIOBST[5]	IOFS

注 **1：** 在本例中指的是 31 kHz INTRC 时钟源。

2： 当从休眠模式和所有空闲模式唤醒时都需 TCSD （表 28-12 中参数 38）延时，此延时与其他任何所需的延时同时进行 （见第 **3.4** 节 **“空闲模式”**）。

3： 包括 INTOSC 8 MHz 时钟源和后分频器产生的频率。

4： TOST 是振荡器起振定时器周期 （表 28-12 中参数 32）。 t_{rc} 是 PLL 锁定延时定时器的延迟时间 （表 28-9 中参数 F12），后者也被称为 TPLL。

5： 在 INTOSC 稳定时间 TIOBST （表 28-12 中参数 39）延时期间，代码继续执行。

 初稿

4.0 复位

PIC18F2455/2550/4455/4550 器件有以下几种不同的复位方式：

a) 上电复位 （POR）
b) 正常工作状态下的 $\overline{MCLR}$ 复位
c) 功耗管理模式下的 $\overline{MCLR}$ 复位
d) 看门狗定时器 （WDT）复位 （执行程序期间）
e) 可编程欠压复位 （BOR）
f) RESET 指令
g) 堆栈满复位
h) 堆栈下溢复位

本节将讨论由 $\overline{MCLR}$、POR 和 BOR 产生的各种复位以及各种起振定时器的操作。堆栈复位事件将在第 **5.1.2.4 节"堆栈满和下溢复位"**中讨论。WDT 复位将在第 **25.2 节 "看门狗定时器 （WDT）"** 中讨论。

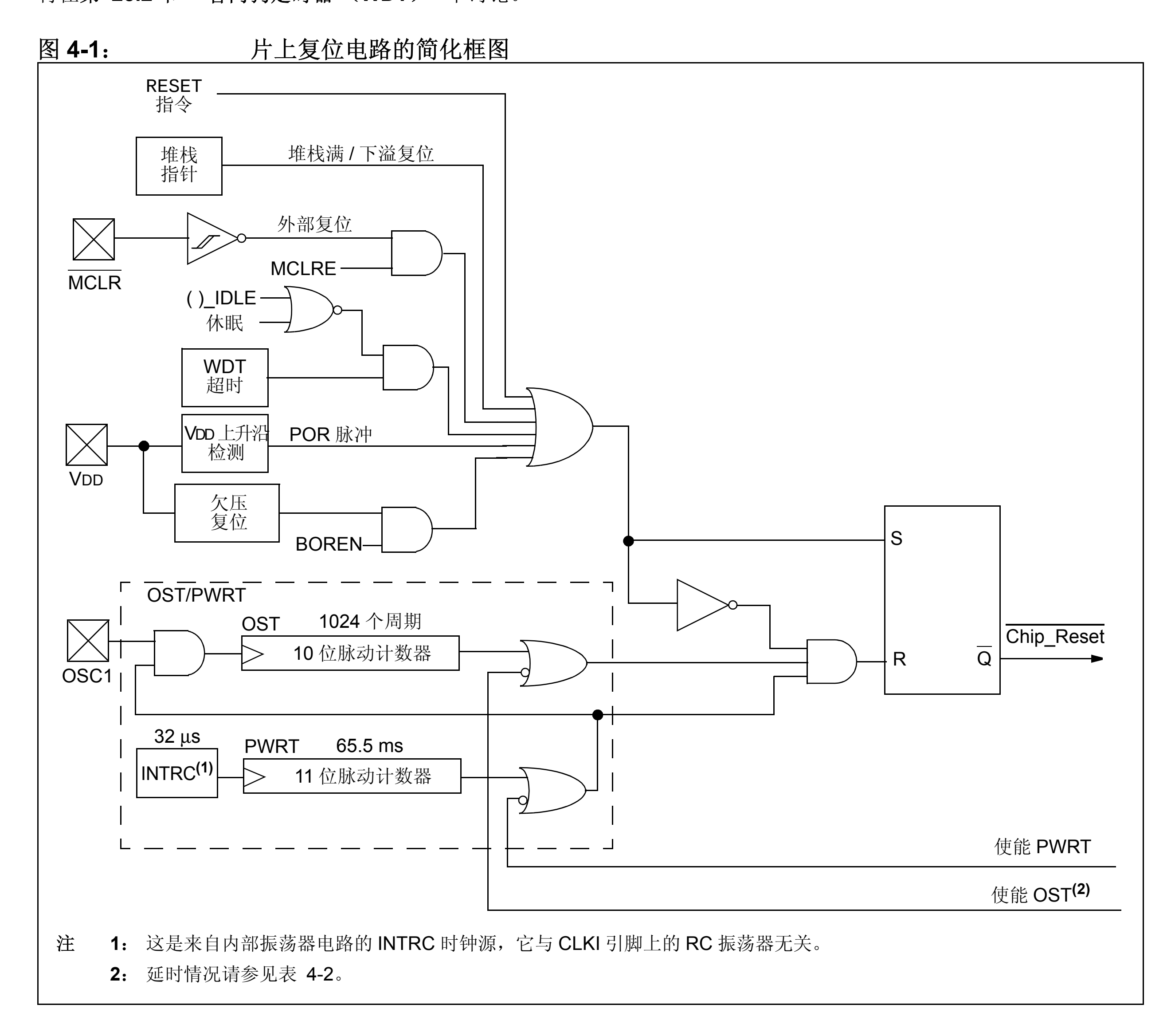
图 4-1 给出了片上复位电路的简化框图。

4.1 RCON 寄存器

可通过 RCON 寄存器 （寄存器 4-1）跟踪器件复位事件。该寄存器的低 5 位表示特定的复位事件。在大部分情况下，只能通过事件将这些位清零，而且必须在随后的应用程序中将它们置 1。读这 5 位标志位可以知道刚刚发生过的复位的类型。第 **4.6 节 "寄存器的复位状态"**中对此进行了更详细的说明。

RCON 寄存器中还有用于设置中断优先级的控制位（IPEN）和用于对 BOR 进行软件控制的控制位（SBOREN）。第 **9.0 节 "中断"**将讨论中断优先级。第 **4.4 节 "欠压复位 （BOR）"** 将讨论 BOR。

图 **4-1**： 片上复位电路的简化框图



注 1： 这是来自内部振荡器电路的 INTRC 时钟源，它与 CLKI 引脚上的 RC 振荡器无关。
2： 延时情况请参见表 4-2。

寄存器 4-1： RCON：复位控制寄存器

R/W-0	R/W-1[1]	U-0	R/W-1	R-1	R-1	R/W-0[2]	R/W-0
IPEN	SBOREN	—	$\overline{\text{RI}}$	$\overline{\text{TO}}$	$\overline{\text{PD}}$	$\overline{\text{POR}}$	$\overline{\text{BOR}}$
bit 7							bit 0

bit 7 **IPEN**：中断优先级使能位
1 = 使能中断优先级
0 = 禁止中断优先级 （PIC16CXXX 兼容模式）

bit 6 **SBOREN**：BOR 软件使能位 [1]
如果 BOREN1:BOREN0 = 01：
1 = 使能 BOR
0 = 禁止 BOR
如果 BOREN1:BOREN0 = 00、 10 或 11：
该位被禁止并读为 0。

bit 5 未用：读为 0。

bit 4 **$\overline{\text{RI}}$**：RESET 指令标志位
1 = 未执行 RESET 指令 （只能由固件置 1）
0 = 已执行 RESET 指令，导致器件复位 （必须在欠压复位发生后由软件置 1）

bit 3 **$\overline{\text{TO}}$**：看门狗超时标志位
1 = 通过上电、 CLRWDT 指令或 SLEEP 指令置 1
0 = 发生了 WDT 超时

bit 2 **$\overline{\text{PD}}$**：掉电检测标志位
1 = 通过上电或 CLRWDT 指令置 1
0 = 通过执行 SLEEP 指令置 1

bit 1 **$\overline{\text{POR}}$**：上电复位状态位 [2]
1 = 未发生上电复位 （只能由固件置 1）
0 = 已发生上电复位 （必须在发生上电复位后由软件置 1）

bit 0 **$\overline{\text{BOR}}$** ：欠压复位状态位
1 = 未发生欠压复位 （只能由固件置 1）
0 = 发生了欠压复位 （必须在欠压复位发生之后由软件置 1）

注 **1**：如果使能 SBOREN 位，其复位状态为 1 ；否则为 0。
2：$\overline{\text{POR}}$ 的实际复位值由器件复位的类型决定。欲知更多信息，请参见本寄存器说明后面的 “注”和第 **4.6** 节 **“寄存器的复位状态”**。

图注：			
R = 可读位	W = 可写位	U = 未用位，读为 0	
-n = 上电复位时的值	1 = 置 1	0 = 清零	x = 未知

> 注 **1**：建议在检测到上电复位后，将 $\overline{\text{POR}}$ 位置 1，以便继续检测后续的上电复位。
> **2**：当 $\overline{\text{BOR}}$ 为 0 且 $\overline{\text{POR}}$ 为 1 时 （假定在上电复位之后立即由软件将 $\overline{\text{POR}}$ 置 1），称已发生了欠压复位。

4.2 主复位（$\overline{MCLR}$）

$\overline{MCLR}$ 引脚提供了用外部硬件触发器件复位的方法。将该引脚拉低可以产生复位信号。这些器件在 MCLR 复位路径上有一个噪声滤波器，该滤波器检测并滤除小的脉冲。

任何内部复位，包括 WDT 复位，都不能将 $\overline{MCLR}$ 引脚驱动为低电平。

在 PIC18F2455/2550/4455/4550 器件中，可以用 MCLRE 配置位禁止 MCLR 输入。当禁止 MCLR 时，该引脚成为一个数字输入引脚。如需更多信息，请参见**第 10.5 节 "PORTE、TRISE 和 LATE 寄存器"**。

4.3 上电复位（POR）

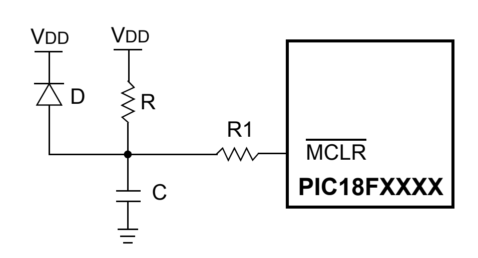
只要当 VDD 超过设定的门限值后，就会在片上产生上电复位脉冲。这使得 VDD 达到满足器件正常工作的值时，器件会以初始化状态启动。

要使用 POR 电路，可以将 $\overline{MCLR}$ 引脚通过一个电阻（阻值为 1 kΩ 到 10 kΩ）连接到 VDD。这样可以省去产生上电复位延时通常所需的外部 RC 元件。 VDD 的最小上升速率已规定（参数 D004，**第 28.1 节 "直流特性"**）。对于上升速率缓慢的情况，请参见图 4-2。

当器件开始正常工作（即退出复位状态）时，必须满足特定的工作参数要求（电压、频率和温度等），才能确保其正常工作。如果不满足这些条件，那么器件必须保持在复位状态，直到满足工作条件为止。

上电复位事件由 $\overline{POR}$ 位（RCON<1>）捕获。只要发生上电复位，此位的状态就被置为 `0`，任何其他复位事件均不能改变它。POR 不能被硬件复位为 `1`。要捕获多个事件，用户必须在 POR 之后用软件将该位复位为 `1`。

图 4-2： 外部上电复位电路（VDD 慢速上电）



注 1： 仅当 VDD 上电速率过慢时才需要外部上电复位电路。当 VDD 掉电时，二极管 D 帮助电容迅速放电。

2： 建议 R < 40 kΩ，以确保电阻 R 两端压降符合器件的电气规范。

3： R1 ≥ 1 kΩ 将限制电流从外部电容 C 流入 MCLR，以避免由于静电放电（Electrostatic Discharge， ESD）或电超载（Electrical Overstress， EOS）导致 $\overline{MCLR}$/VPP 引脚损坏。

4.4 欠压复位 （BOR）

PIC18F2455/2550/4455/4550 器件包含一个 BOR 电路，并为用户提供多个配置和节能选项。 BOR 由 BORV1:BORV0 和 BOREN1:BOREN0 配置位控制。总共有四种 BOR 配置，归纳在表 4-1 中。

BOR 门限值由 BORV1:BORV0 位设置。如果使能 BOR （BOREN1:BOREN0 为非零值），当 V_{DD} 跌落到低于 V_{BOR} （参数 D005，第 **28.1 节 "直流特性"**）的时间超过 T_{BOR} （参数 35，表 28-12）时就会使器件复位。如果 V_{DD} 跌落到 V_{BOR} 以下的时间小于 T_{BOR}，可能不一定发生复位。芯片将保持在欠压复位状态，直至 V_{DD} 电压上升到 V_{BOR} 以上。

如果使能上电延时定时器，则它将在 V_{DD} 上升到超过 V_{BOR} 之后开始工作；并使芯片在延时 T_{PWRT} （参数 33，表 28-12）期间保持复位。如果在上电延时定时器运行过程中 V_{DD} 电压降到 V_{BOR} 以下，芯片将重新回到欠压复位状态并初始化上电延时定时器。一旦 V_{DD} 电压超过 V_{BOR}，上电延时定时器将重新执行延时操作。

BOR 和上电延时定时器 （PWRT）是分别配置的。使能 BOR 复位并不会自动使能 PWRT。

4.4.1 用软件使能 BOR

当 BOREN1:BOREN0 = `01` 时，用户可以用软件使能或禁止 BOR。这可通过控制位 SBOREN（RCON<6>）完成。如前所述，将 SBOREN 置 1 可使能 BOR。清零 SBOREN 将完全禁止 BOR。 SBOREN 位只用于该模式，其他情况下它将读为 `0`。

用软件控制BOR位可使用户能更灵活地定制应用程序，而无需通过对器件重新编程来更改 BOR 配置。它还允许用户通过减少 BOR 消耗的电流，用软件调节器件的功耗。虽然 BOR 的电流通常很小，但可能会对低功耗应用有一些影响。

> **注：** 即使当 BOR 受软件控制时， BOR 复位电平仍将由 BORV1:BORV0 配置位设置。该值不能用软件更改。

4.4.2 检测 BOR

使能 BOR 后，当发生 BOR 或 POR 事件时，$\overline{BOR}$ 位总是复位为 `0`。因此仅通过读 $\overline{BOR}$ 的状态很难判断是否发生了 BOR 事件。更可靠的方法是同时检查 $\overline{POR}$ 和 $\overline{BOR}$ 的状态。假定在发生 POR 事件后，立即用软件将 $\overline{POR}$ 位置 `1`。如果 $\overline{BOR}$ 为 `0` 而 $\overline{POR}$ 为 `1`，就可以明确地断定已经发生了 BOR 事件。

4.4.3 在休眠模式下禁止 BOR

当 BOREN1:BOREN0 = `10` 时，BOR 保持受硬件控制状态并且像前面描述的那样工作。每当器件进入休眠模式时，就会自动禁止 BOR。当器件返回到其他工作模式时，又将自动重新使能 BOR。

此模式使应用程序能在有效地执行代码的同时从欠压状态恢复，这也是器件最需要 BOR 保护的时刻。同时，通过消除增加的 BOR 电流，可以节省休眠模式下的额外功耗。

表 4-1： BOR 配置

BOR 配置		SBOREN （RCON<6>）的状态	BOR 操作
BOREN1	BOREN0		
0	0	不可用	禁止 BOR ；必须通过对配置位重新编程才能使能 BOR。
0	1	可用	用软件使能 BOR ；工作模式由 SBOREN 控制。
1	0	不可用	在运行和空闲模式下用硬件使能 BOR，在休眠模式下禁止 BOR。
1	1	不可用	用硬件使能 BOR ；必须通过对配置位重新编程才能禁止 BOR。

4.5 器件复位定时器

PIC18F2455/2550/4455/4550 器件包含了三个独立的片上定时器，帮助调节上电复位过程。它们的主要功能是确保在代码执行之前系统时钟是稳定的。这些定时器是：

- 上电延时定时器 （PWRT）
- 振荡器起振定时器 （OST）
- PLL 锁定延时定时器

4.5.1 上电延时定时器 （PWRT）

PIC18F2455/2550/4455/4550 器件的上电延时定时器（PWRT）是一个 11 位计数器，使用 INTRC 时钟源作为时钟输入。产生的延迟时间约为 2048 x 32 μs = 65.6 ms。当 PWRT 计数时，器件保持在复位状态。

上电延迟时间取决于 INTRC 时钟，并且由于温度和工艺的不同，芯片的延迟时间也各不相同。详情请参见直流参数 33 （表 28-12）。

通过将 $\overline{\text{PWRTEN}}$ 配置位清零可使能 PWRT。

4.5.2 振荡器起振定时器 （OST）

在 PWRT 延时 （参数 33，表 28-12）结束以后，由振荡器起振定时器 （OST）提供一个 1024 个振荡周期的延时 （从 OSC1 输入），从而确保晶振或谐振器已经起振并稳定工作。

只有在 XT、 HS 和 HSPLL 模式下，且仅当发生上电复位或从大多数功耗管理模式退出时，才启动 OST 延时。

4.5.3 PLL 锁定延时

当在 PLL 模式下使能 PLL 时，上电复位后的延时时序与其他振荡模式略有不同。使用一个独立的定时器来提供一段足够让 PLL 锁定主振荡器频率的固定延时。PLL 锁定延时 （TPLL）通常为 2 ms，且在振荡器起振延时后发生。

4.5.4 延时时序

上电延时时序如下：

1. POR 条件清除后，启动 PWRT 延时 （如果使能）。
2. 然后， OST 被激活。

总延迟时间取决于振荡器的配置和 PWRT 的状态。图 4-3、图 4-4、图 4-5、图 4-6和图 4-7各自描述了不同的上电延时时序，其中上电延时定时器均被使能，并且器件工作在 HS 振荡模式下。图 4-3 到 4-6 也适用于在 XT 模式下工作的器件。对于工作在 RC 模式下并禁止 PWRT 的器件，将没有延时。

由于延时是由 POR 脉冲触发的，因此如果 $\overline{\text{MCLR}}$ 保持足够长时间的低电平，所有延时都将无效。将 $\overline{\text{MCLR}}$ 电平拉高后程序将立即开始执行代码 （图 4-5）。这对于测试或同步多个并行工作的 PIC18FXXXX 器件是非常有用的。

表 4-2： 不同情况下的延时

振荡器配置	上电复位 (2) 和欠压复位		从功耗管理模式退出
	$\overline{\text{PWRTEN}}$ = 0	$\overline{\text{PWRTEN}}$ = 1	
HS 和 XT	66 ms(1) + 1024 TOSC	1024 TOSC	1024 TOSC
HSPLL 和 XTPLL	66 ms(1) + 1024 TOSC + 2 ms(2)	1024 TOSC + 2 ms(2)	1024 TOSC + 2 ms(2)
EC 和 ECIO	66 ms(1)	—	—
ECPLL 和 ECPIO	66 ms(1) + 2 ms(2)	2 ms(2)	2 ms(2)
INTIO 和 INTCKO	66 ms(1)	—	—
INTHS 和 INTXT	66 ms(1) + 1024 TOSC	1024 TOSC	1024 TOSC

注 1： 66 ms （65.5 ms）是上电延时定时器 （PWRT）延迟时间的标称值。
2： 2 ms 是 PLL 锁定所需时间的标称值。

图 4-3： 上电延时时序 （MCLR 连接到 VDD，VDD 电压上升时间 < TPWRT）

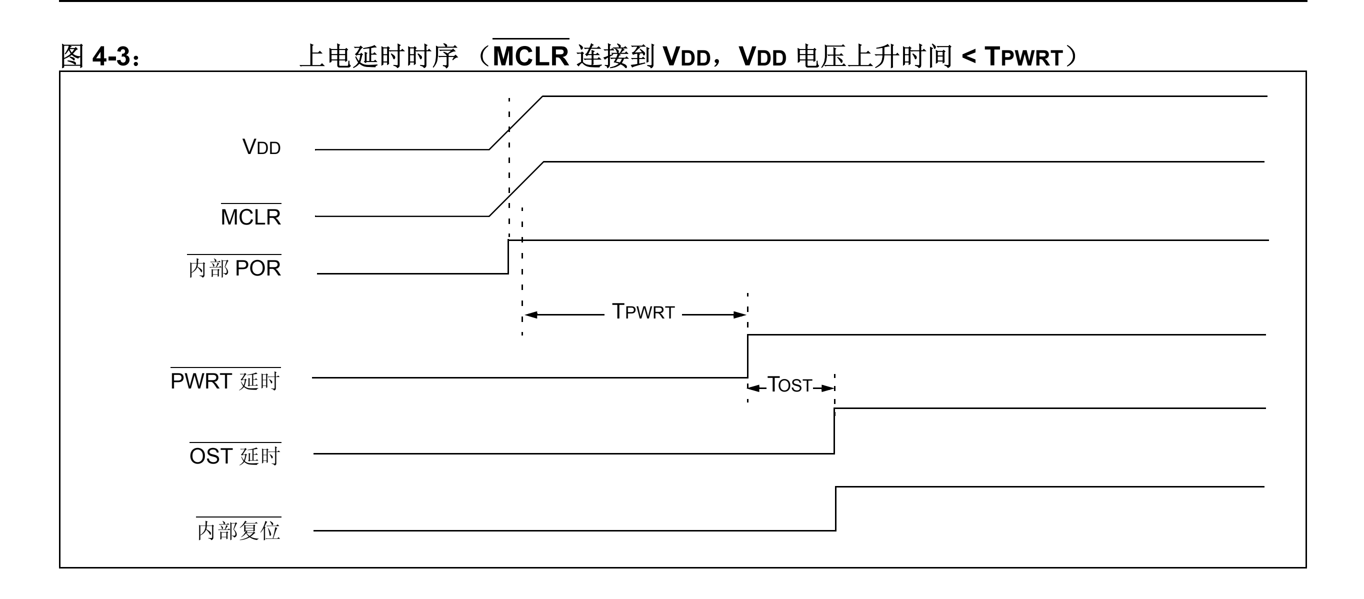


图 4-4： 上电延时时序 （MCLR 未连接到 VDD）：情形 1

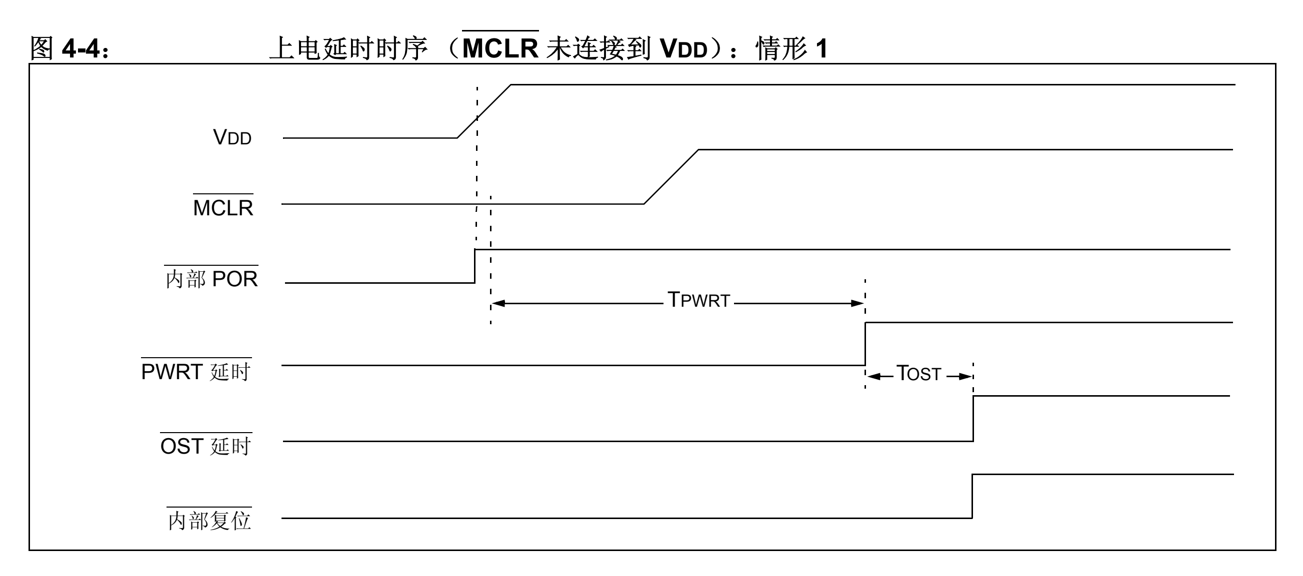
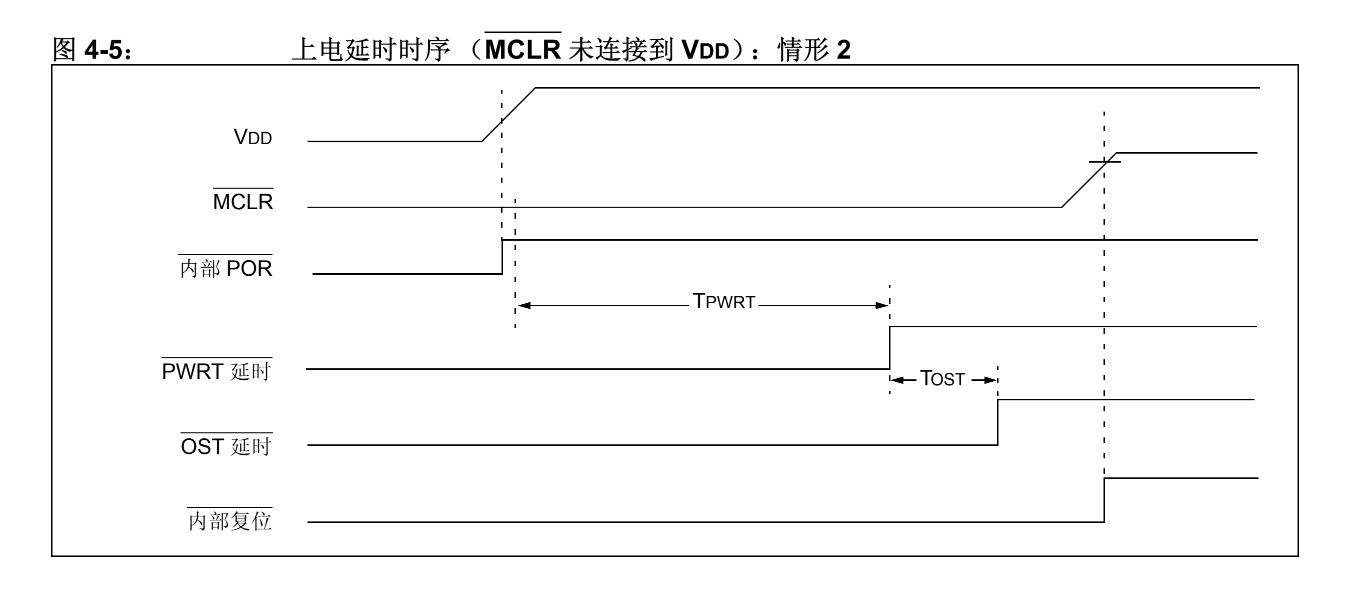


图 4-5： 上电延时时序 （MCLR 未连接到 VDD）：情形 2



 初稿

图 4-6： 缓慢上升时间 （$\overline{MCLR}$ 连接到 VDD，VDD 上升时间 > TPWRT）

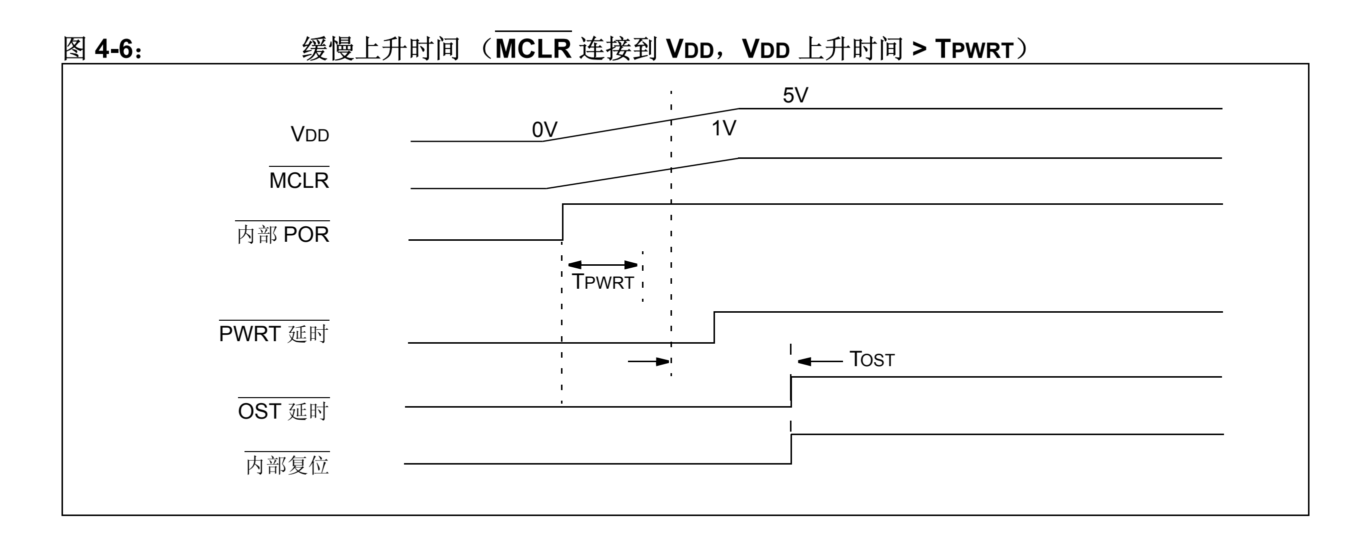
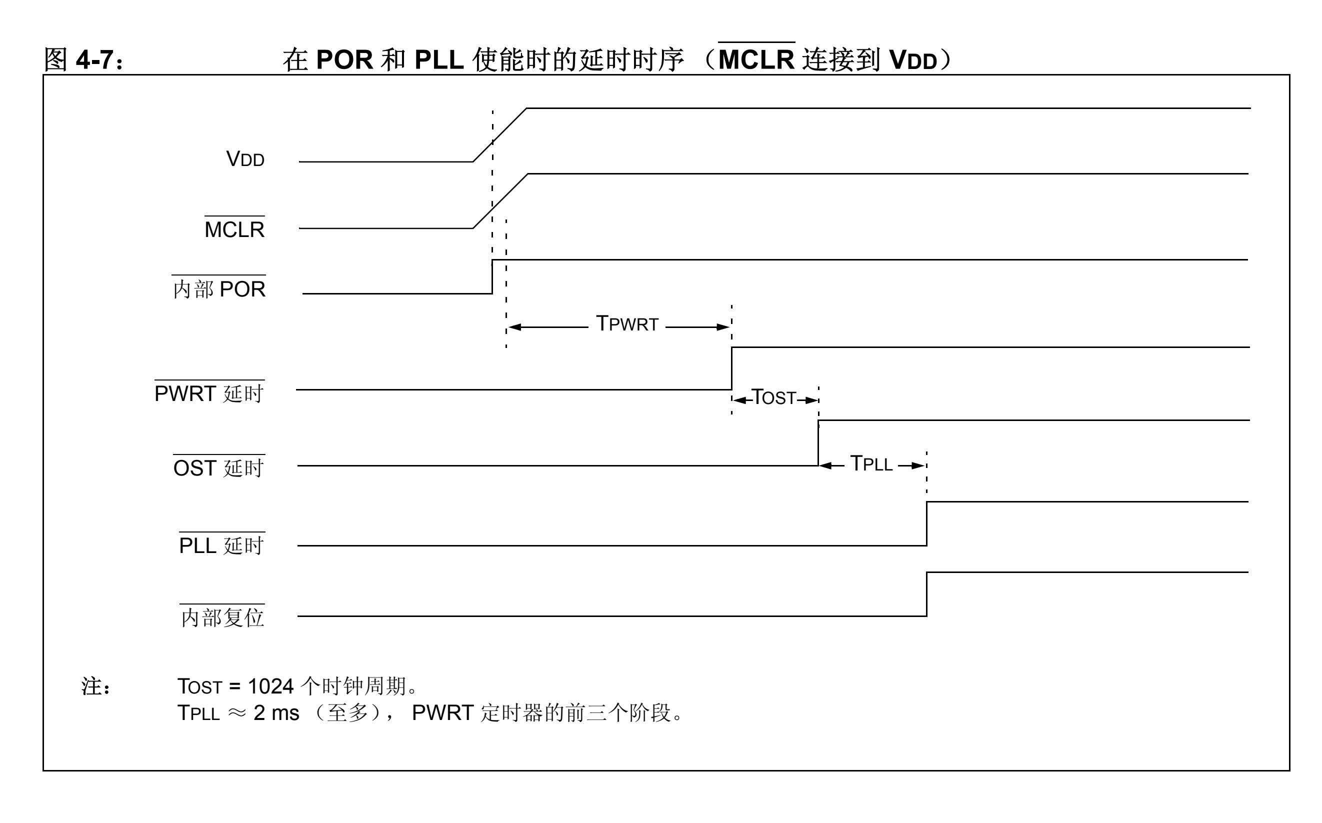


图 4-7： 在 POR 和 PLL 使能时的延时时序 （$\overline{MCLR}$ 连接到 VDD）



注： TOST = 1024 个时钟周期。
TPLL ≈ 2 ms （至多）， PWRT 定时器的前三个阶段。

4.6 寄存器的复位状态

大多数寄存器不受复位的影响。在 POR 时这些寄存器的状态不确定，而在其他复位时它们的状态不变。而其他寄存器则根据不同的复位类型被强制为 “复位状态”。

因为 WDT 唤醒被视为恢复正常的工作，所以大部分寄存器不受 WDT 唤醒的影响。RCON 寄存器中的状态位：RI、TO、PD、POR 和 BOR 在不同的复位情形中会分别被置 1 或清零，如表 4-3 所示。可在软件中使用这些状态位来判断复位的类型。

表 4-4 给出了所有特殊功能寄存器的复位状态。这些复位被分类为上电和欠压复位、主复位和 WDT 复位以及 WDT 唤醒复位。

表 4-3： RCON 寄存器的状态位、含义以及初始化状态

条件	程序计数器	RCON 寄存器						STKPTR 寄存器	
		SBOREN	$\overline{RI}$	$\overline{TO}$	$\overline{PD}$	$\overline{POR}$	$\overline{BOR}$	STKFUL	STKUNF
上电复位	0000h	1	1	1	1	0	0	0	0
RESET 指令	0000h	u(2)	0	u	u	u	u	u	u
欠压	0000h	u(2)	1	1	1	u	0	u	u
功耗管理运行模式期间的 $\overline{MCLR}$	0000h	u(2)	u	1	u	u	u	u	u
功耗管理空闲和休眠模式期间的 $\overline{MCLR}$	0000h	u(2)	u	1	0	u	u	u	u
全功耗或功耗管理运行模式期间的 WDT 超时	0000h	u(2)	u	0	u	u	u	u	u
全功耗执行期间的 $\overline{MCLR}$	0000h	u(2)	u	u	u	u	u	u	u
堆栈满复位 （STVREN = 1）	0000h	u(2)	u	u	u	u	u	1	u
堆栈下溢复位 （STVREN = 1）	0000h	u(2)	u	u	u	u	u	u	1
堆栈下溢错误 （不是真正的复位， STVREN = 0）	0000h	u(2)	u	u	u	u	u	u	1
功耗管理空闲或休眠模式期间的 WDT 超时	PC + 2	u(2)	u	0	0	u	u	u	u
通过中断从功耗管理模式退出	PC + 2	u(2)	u	u	0	u	u	u	u

图注： u = 不变

注 1： 当器件被中断唤醒且 GIEH 或 GIEL 位置 1 时， PC 装入中断向量 （0008h 或 0018h）。

2： 当使能软件 BOR 时 （BOREN1:BOREN0 配置位 = 01 且 SBOREN = 1）， POR 位的复位状态是 1 且其他复位不能改变该状态。否则，复位状态是 0。

表 4-4： 所有寄存器的初始化状态

寄存器	适用器件				上电复位和欠压复位	MCLR 复位、WDT 复位、RESET 指令和堆栈复位	通过 WDT 或中断唤醒器件
TOSU	2455	2550	4455	4550	---0 0000	---0 0000	---0 uuuu(1)
TOSH	2455	2550	4455	4550	0000 0000	0000 0000	uuuu uuuu(1)
TOSL	2455	2550	4455	4550	0000 0000	0000 0000	uuuu uuuu(1)
STKPTR	2455	2550	4455	4550	00-0 0000	uu-0 0000	uu-u uuuu(1)
PCLATU	2455	2550	4455	4550	---0 0000	---0 0000	---u uuuu
PCLATH	2455	2550	4455	4550	0000 0000	0000 0000	uuuu uuuu
PCL	2455	2550	4455	4550	0000 0000	0000 0000	PC + 2(2)
TBLPTRU	2455	2550	4455	4550	--00 0000	--00 0000	--uu uuuu
TBLPTRH	2455	2550	4455	4550	0000 0000	0000 0000	uuuu uuuu
TBLPTRL	2455	2550	4455	4550	0000 0000	0000 0000	uuuu uuuu
TABLAT	2455	2550	4455	4550	0000 0000	0000 0000	uuuu uuuu
PRODH	2455	2550	4455	4550	xxxx xxxx	uuuu uuuu	uuuu uuuu
PRODL	2455	2550	4455	4550	xxxx xxxx	uuuu uuuu	uuuu uuuu
INTCON	2455	2550	4455	4550	0000 000x	0000 000u	uuuu uuuu(3)
INTCON2	2455	2550	4455	4550	1111 -1-1	1111 -1-1	uuuu -u-u(3)
INTCON3	2455	2550	4455	4550	11-0 0-00	11-0 0-00	uu-u u-uu(3)
INDF0	2455	2550	4455	4550	N/A	N/A	N/A
POSTINC0	2455	2550	4455	4550	N/A	N/A	N/A
POSTDEC0	2455	2550	4455	4550	N/A	N/A	N/A
PREINC0	2455	2550	4455	4550	N/A	N/A	N/A
PLUSW0	2455	2550	4455	4550	N/A	N/A	N/A
FSR0H	2455	2550	4455	4550	---- 0000	---- 0000	---- uuuu
FSR0L	2455	2550	4455	4550	xxxx xxxx	uuuu uuuu	uuuu uuuu
WREG	2455	2550	4455	4550	xxxx xxxx	uuuu uuuu	uuuu uuuu
INDF1	2455	2550	4455	4550	N/A	N/A	N/A
POSTINC1	2455	2550	4455	4550	N/A	N/A	N/A
POSTDEC1	2455	2550	4455	4550	N/A	N/A	N/A
PREINC1	2455	2550	4455	4550	N/A	N/A	N/A
PLUSW1	2455	2550	4455	4550	N/A	N/A	N/A
FSR1H	2455	2550	4455	4550	---- 0000	---- 0000	---- uuuu
FSR1L	2455	2550	4455	4550	xxxx xxxx	uuuu uuuu	uuuu uuuu
BSR	2455	2550	4455	4550	---- 0000	---- 0000	---- uuuu

图注： u = 不变，x = 未知，— = 未用 （读为 0），q = 取值依情况而定。
阴影单元表示不适用于指定器件。

注 1： 当器件被中断唤醒且 GIEL 或 GIEH 位被置 1 时，用 PC 的当前值更新 TOSU、 TOSH 和 TOSL。将 STKPTR 修改为指向硬件堆栈的下一个单元。

2： INTCONx 或 PIRx 寄存器中的一位或多位会受到影响 （引起唤醒）。

3： 当器件被中断唤醒且 GIEL 或 GIEH 位被置 1 时， PC 装入中断向量 （0008h 或 0018h）。

4： 具体条件下的复位值，请参见表 4-3。

5： 根据选定的振荡模式使能 PORTA<6>、LATA<6> 和 TRISA<6>。若未将这些位所对应的引脚配置为 PORTA 引脚，则它们将被禁止且读为 0。

表 4-4： 所有寄存器的初始化状态 （续）

寄存器	适用器件				上电复位和 欠压复位	MCLR 复位、 WDT 复位、 RESET 指令和 堆栈复位	通过 WDT 或中断唤醒器件
INDF2	2455	2550	4455	4550	N/A	N/A	N/A
POSTINC2	2455	2550	4455	4550	N/A	N/A	N/A
POSTDEC2	2455	2550	4455	4550	N/A	N/A	N/A
PREINC2	2455	2550	4455	4550	N/A	N/A	N/A
PLUSW2	2455	2550	4455	4550	N/A	N/A	N/A
FSR2H	2455	2550	4455	4550	---- 0000	---- 0000	---- uuuu
FSR2L	2455	2550	4455	4550	xxxx xxxx	uuuu uuuu	uuuu uuuu
STATUS	2455	2550	4455	4550	---x xxxx	---u uuuu	---u uuuu
TMR0H	2455	2550	4455	4550	0000 0000	0000 0000	uuuu uuuu
TMR0L	2455	2550	4455	4550	xxxx xxxx	uuuu uuuu	uuuu uuuu
T0CON	2455	2550	4455	4550	1111 1111	1111 1111	uuuu uuuu
OSCCON	2455	2550	4455	4550	0100 q000	0100 00q0	uuuu uuqu
HLVDCON	2455	2550	4455	4550	0-00 0101	0-00 0101	u-uu uuuu
WDTCON	2455	2550	4455	4550	---- ---0	---- ---0	---- ---u
RCON(4)	2455	2550	4455	4550	0q-1 11q0	0q-q qquu	uq-u qquu
TMR1H	2455	2550	4455	4550	xxxx xxxx	uuuu uuuu	uuuu uuuu
TMR1L	2455	2550	4455	4550	xxxx xxxx	uuuu uuuu	uuuu uuuu
T1CON	2455	2550	4455	4550	0000 0000	u0uu uuuu	uuuu uuuu
TMR2	2455	2550	4455	4550	0000 0000	0000 0000	uuuu uuuu
PR2	2455	2550	4455	4550	1111 1111	1111 1111	1111 1111
T2CON	2455	2550	4455	4550	-000 0000	-000 0000	-uuu uuuu
SSPBUF	2455	2550	4455	4550	xxxx xxxx	uuuu uuuu	uuuu uuuu
SSPADD	2455	2550	4455	4550	0000 0000	0000 0000	uuuu uuuu
SSPSTAT	2455	2550	4455	4550	0000 0000	0000 0000	uuuu uuuu
SSPCON1	2455	2550	4455	4550	0000 0000	0000 0000	uuuu uuuu
SSPCON2	2455	2550	4455	4550	0000 0000	0000 0000	uuuu uuuu
ADRESH	2455	2550	4455	4550	xxxx xxxx	uuuu uuuu	uuuu uuuu
ADRESL	2455	2550	4455	4550	xxxx xxxx	uuuu uuuu	uuuu uuuu
ADCON0	2455	2550	4455	4550	--00 0000	--00 0000	--uu uuuu
ADCON1	2455	2550	4455	4550	--00 0qqq	--00 0qqq	--uu uuuu
ADCON2	2455	2550	4455	4550	0-00 0000	0-00 0000	u-uu uuuu

图注： u = 不变，x = 未知， － = 未用 （ 读为 0），q = 取值依情况而定。
阴影单元表示不适用于指定器件。

注 1： 当器件被中断唤醒且 GIEL 或 GIEH 位被置 1 时，用 PC 的当前值更新 TOSU、 TOSH 和 TOSL。将 STKPTR 修改为指向硬件堆栈的下一个单元。

2： INTCONx 或 PIRx 寄存器中的一位或多位会受到影响 （引起唤醒）。

3： 当器件被中断唤醒且 GIEL 或 GIEH 位被置 1 时， PC 装入中断向量 （0008h 或 0018h）。

4： 具体条件下的复位值，请参见表 4-3。

5： 根据选定的振荡模式使能 PORTA<6>、LATA<6> 和 TRISA<6>。若未将这些位所对应的引脚配置为 PORTA 引脚，则它们将被禁止且读为 0。

表 **4-4**： 所有寄存器的初始化状态 （续）

寄存器	适用器件				上电复位和欠压复位	MCLR 复位、WDT 复位、RESET 指令和堆栈复位	通过 WDT 或中断唤醒器件
CCPR1H	2455	2550	4455	4550	xxxx xxxx	uuuu uuuu	uuuu uuuu
CCPR1L	2455	2550	4455	4550	xxxx xxxx	uuuu uuuu	uuuu uuuu
CCP1CON	2455	2550	4455	4550	--00 0000	--00 0000	--uu uuuu
	2455	2550	4455	4550	0000 0000	0000 0000	uuuu uuuu
CCPR2H	2455	2550	4455	4550	xxxx xxxx	uuuu uuuu	uuuu uuuu
CCPR2L	2455	2550	4455	4550	xxxx xxxx	uuuu uuuu	uuuu uuuu
CCP2CON	2455	2550	4455	4550	0000 0000	0000 0000	uuuu uuuu
BAUDCON	2455	2550	4455	4550	01-0 0-00	01-0 0-00	uu-u u-uu
ECCP1DEL	2455	2550	4455	4550	0000 0000	0000 0000	uuuu uuuu
ECCP1AS	2455	2550	4455	4550	0000 0000	0000 0000	uuuu uuuu
CVRCON	2455	2550	4455	4550	0000 0000	0000 0000	uuuu uuuu
CMCON	2455	2550	4455	4550	0000 0111	0000 0111	uuuu uuuu
TMR3H	2455	2550	4455	4550	xxxx xxxx	uuuu uuuu	uuuu uuuu
TMR3L	2455	2550	4455	4550	xxxx xxxx	uuuu uuuu	uuuu uuuu
T3CON	2455	2550	4455	4550	0000 0000	uuuu uuuu	uuuu uuuu
SPBRGH	2455	2550	4455	4550	0000 0000	0000 0000	uuuu uuuu
SPBRG	2455	2550	4455	4550	0000 0000	0000 0000	uuuu uuuu
RCREG	2455	2550	4455	4550	0000 0000	0000 0000	uuuu uuuu
TXREG	2455	2550	4455	4550	0000 0000	0000 0000	uuuu uuuu
TXSTA	2455	2550	4455	4550	0000 0010	0000 0010	uuuu uuuu
RCSTA	2455	2550	4455	4550	0000 000x	0000 000x	uuuu uuuu
EEADR	2455	2550	4455	4550	0000 0000	0000 0000	uuuu uuuu
EEDATA	2455	2550	4455	4550	0000 0000	0000 0000	uuuu uuuu
EECON2	2455	2550	4455	4550	0000 0000	0000 0000	0000 0000
EECON1	2455	2550	4455	4550	xx-0 x000	uu-0 u000	uu-0 u000

图注： u = 不变，x = 未知，－ = 未用 （读为 0），q = 取值依情况而定。
阴影单元表示不适用于指定器件。

注 **1**： 当器件被中断唤醒且 GIEL 或 GIEH 位被置 1 时，用 PC 的当前值更新 TOSU、 TOSH 和 TOSL。将 STKPTR 修改为指向硬件堆栈的下一个单元。

2： INTCONx 或 PIRx 寄存器中的一位或多位会受到影响 （引起唤醒）。

3： 当器件被中断唤醒且 GIEL 或 GIEH 位被置 1 时， PC 装入中断向量 （0008h 或 0018h）。

4： 具体条件下的复位值，请参见表 4-3。

5： 根据选定的振荡模式使能 PORTA<6>、LATA<6> 和 TRISA<6>。若未将这些位所对应的引脚配置为 PORTA 引脚，则它们将被禁止且读为 0。

表 4-4： 所有寄存器的初始化状态 （续）

寄存器	适用器件				上电复位和 欠压复位	$\overline{\text{MCLR}}$ 复位、 **WDT** 复位、 **RESET** 指令和 堆栈复位	通过 **WDT** 或中断唤醒器件
IPR2	2455	2550	4455	4550	1111 1111	1111 1111	uuuu uuuu
PIR2	2455	2550	4455	4550	0000 0000	0000 0000	uuuu uuuu(3)
PIE2	2455	2550	4455	4550	0000 0000	0000 0000	uuuu uuuu
IPR1	2455	2550	4455	4550	1111 1111	1111 1111	uuuu uuuu
	2455	2550	4455	4550	-111 1111	-111 1111	-uuu uuuu
PIR1	2455	2550	4455	4550	0000 0000	0000 0000	uuuu uuuu(3)
	2455	2550	4455	4550	-000 0000	-000 0000	-uuu uuuu
PIE1	2455	2550	4455	4550	0000 0000	0000 0000	uuuu uuuu
	2455	2550	4455	4550	-000 0000	-000 0000	-uuu uuuu
OSCTUNE	2455	2550	4455	4550	0--0 0000	0--0 0000	u--u uuuu
TRISE	2455	2550	4455	4550	---- -111	---- -111	uuuu -uuu
TRISD	2455	2550	4455	4550	1111 1111	1111 1111	uuuu uuuu
TRISC	2455	2550	4455	4550	11-- -111	11-- -111	uu-- -uuu
TRISB	2455	2550	4455	4550	1111 1111	1111 1111	uuuu uuuu
TRISA(5)	2455	2550	4455	4550	-111 1111(5)	-111 1111(5)	-uuu uuuu(5)
LATE	2455	2550	4455	4550	---- -xxx	---- -uuu	---- -uuu
LATD	2455	2550	4455	4550	xxxx xxxx	uuuu uuuu	uuuu uuuu
LATC	2455	2550	4455	4550	xx-- -xxx	uu-- -uuu	uu-- -uuu
LATB	2455	2550	4455	4550	xxxx xxxx	uuuu uuuu	uuuu uuuu
LATA(5)	2455	2550	4455	4550	-xxx xxxx(5)	-uuu uuuu(5)	-uuu uuuu(5)
PORTE	2455	2550	4455	4550	0--- x000	0--- x000	u--- uuuu
PORTD	2455	2550	4455	4550	xxxx xxxx	uuuu uuuu	uuuu uuuu
PORTC	2455	2550	4455	4550	xxxx -xxx	uuuu -uuu	uuuu -uuu
PORTB	2455	2550	4455	4550	xxxx xxxx	uuuu uuuu	uuuu uuuu
PORTA(5)	2455	2550	4455	4550	-x0x 0000(5)	-u0u 0000(5)	-uuu uuuu(5)

图注： u = 不变，x = 未知，— = 未用 （ 读为 0），q = 取值依情况而定。
阴影单元表示不适用于指定器件。

注 **1**： 当器件被中断唤醒且 GIEL 或 GIEH 位被置 1 时，用 PC 的当前值更新 TOSU、 TOSH 和 TOSL。将 STKPTR 修改为指向硬件堆栈的下一个单元。

2： INTCONx 或 PIRx 寄存器中的一位或多位会受到影响 （引起唤醒）。

3： 当器件被中断唤醒且 GIEL 或 GIEH 位被置 1 时， PC 装入中断向量 （0008h 或 0018h）。

4： 具体条件下的复位值，请参见表 4-3。

5： 根据选定的振荡模式使能 PORTA<6>、LATA<6> 和 TRISA<6>。若未将这些位所对应的引脚配置为 PORTA 引脚，则它们将被禁止且读为 0。

表 4-4： 所有寄存器的初始化状态 （续）

寄存器	适用器件				上电复位和 欠压复位	$\overline{MCLR}$ 复位、 WDT 复位、 RESET 指令和 堆栈复位	通过 WDT 或中断唤醒器件
UEP15	2455	2550	4455	4550	---0 0000	---0 0000	---u uuuu
UEP14	2455	2550	4455	4550	---0 0000	---0 0000	---u uuuu
UEP13	2455	2550	4455	4550	---0 0000	---0 0000	---u uuuu
UEP12	2455	2550	4455	4550	---0 0000	---0 0000	---u uuuu
UEP11	2455	2550	4455	4550	---0 0000	---0 0000	---u uuuu
UEP10	2455	2550	4455	4550	---0 0000	---0 0000	---u uuuu
UEP9	2455	2550	4455	4550	---0 0000	---0 0000	---u uuuu
UEP8	2455	2550	4455	4550	---0 0000	---0 0000	---u uuuu
UEP7	2455	2550	4455	4550	---0 0000	---0 0000	---u uuuu
UEP6	2455	2550	4455	4550	---0 0000	---0 0000	---u uuuu
UEP5	2455	2550	4455	4550	---0 0000	---0 0000	---u uuuu
UEP4	2455	2550	4455	4550	---0 0000	---0 0000	---u uuuu
UEP3	2455	2550	4455	4550	---0 0000	---0 0000	---u uuuu
UEP2	2455	2550	4455	4550	---0 0000	---0 0000	---u uuuu
UEP1	2455	2550	4455	4550	---0 0000	---0 0000	---u uuuu
UEP0	2455	2550	4455	4550	---0 0000	---0 0000	---u uuuu
UCFG	2455	2550	4455	4550	00-0 0000	00-0 0000	uu-u uuuu
UADDR	2455	2550	4455	4550	-000 0000	-000 0000	-uuu uuuu
UCON	2455	2550	4455	4550	-0x0 000-	-0x0 000-	-uuu uuu-
USTAT	2455	2550	4455	4550	-xxx xxx-	-xxx xxx-	-uuu uuu-
UEIE	2455	2550	4455	4550	0--0 0000	0--0 0000	u--u uuuu
UEIR	2455	2550	4455	4550	0--0 0000	0--0 0000	u--u uuuu
UIE	2455	2550	4455	4550	-000 0000	-000 0000	-uuu uuuu
UIR	2455	2550	4455	4550	-000 0000	-000 0000	-uuu uuuu
UFRMH	2455	2550	4455	4550	---- -xxx	---- -xxx	---- -uuu
UFRML	2455	2550	4455	4550	xxxx xxxx	xxxx xxxx	uuuu uuuu
SPPCON	2455	2550	4455	4550	---- --00	---- --00	---- --uu
SPPEPS	2455	2550	4455	4550	00-0 0000	00-0 0000	uu-u uuuu
SPPCFG	2455	2550	4455	4550	0000 0000	0000 0000	uuuu uuuu
SPPDATA	2455	2550	4455	4550	0000 0000	0000 0000	uuuu uuuu

图注： u = 不变，x = 未知，— = 未用 （读为 0），q = 取值依情况而定。
阴影单元表示不适用于指定器件。

注 1： 当器件被中断唤醒且 GIEL 或 GIEH 位被置 1 时，用 PC 的当前值更新 TOSU、 TOSH 和 TOSL。将 STKPTR 修改为指向硬件堆栈的下一个单元。

2： INTCONx 或 PIRx 寄存器中的一位或多位会受到影响 （引起唤醒）。

3： 当器件被中断唤醒且 GIEL 或 GIEH 位被置 1 时， PC 装入中断向量 （0008h 或 0018h）。

4： 具体条件下的复位值，请参见表 4-3。

5： 根据选定的振荡模式使能 PORTA<6>、LATA<6> 和 TRISA<6>。若未将这些位所对应的引脚配置为 PORTA 引脚，则它们将被禁止且读为 0。

注：

5.0 存储器构成

PIC18 增强型单片机有 3 种类型的存储器：

- 程序存储器
- 数据 RAM
- 数据 EEPROM

在哈佛架构的器件中，数据和程序存储器使用不同的总线，因而可同时访问这两种存储空间。数据 EEPROM，从实际用途而言，可以被视为外设，因为它是通过一组控制寄存器来寻址和访问的。

第 **6.0** 节 **“闪存程序存储器”**提供了闪存程序存储器操作的详细信息。第 **7.0** 节 **“数据 EEPROM 存储器”**中将单独讨论数据 EEPROM。

5.1 程序存储器构成

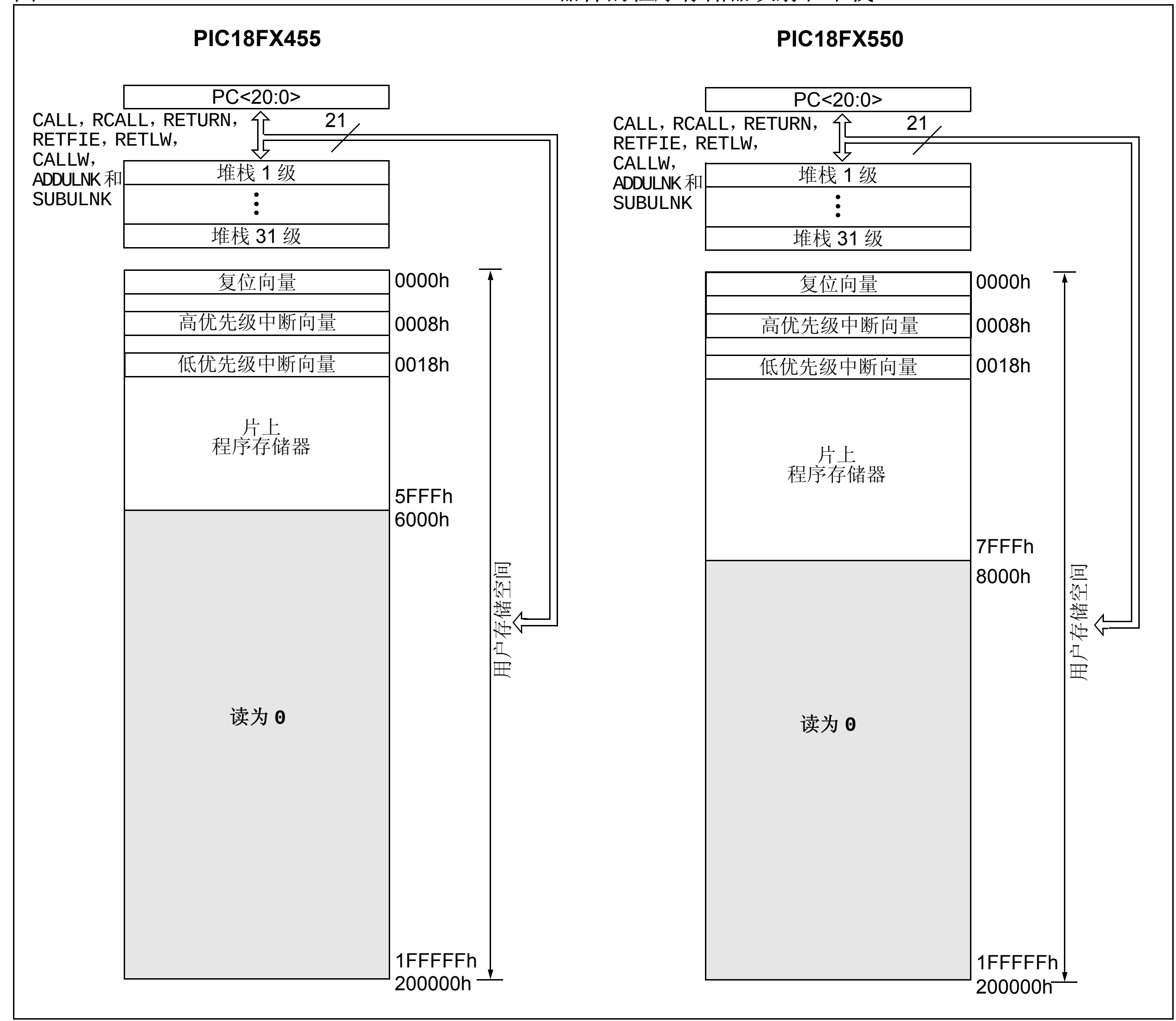
PIC18 单片机具有 21 位程序计数器，可以对 2 MB 的程序存储空间进行寻址。访问存储器物理地址上边界和 2 MB 地址之间的存储单元将会返回全 `0` （`NOP` 指令）。

PIC18F2455 和 PIC18F4455 具有 24 KB 的闪存存储器，能够存储多达 12,288 条单字指令。PIC18F2550 和 PIC18F4550 具有 32 KB 的闪存存储器，能够存储多达 16,384 条单字指令。

PIC18 器件有两个中断向量。复位向量地址为 0000h，中断向量地址为 0008h 和 0018h。

图 5-1 所示为 PIC18FX455 和 PIC18FX550 器件的程序存储器映射。

图 5-1： PIC18F2455/2550/4455/4550 器件的程序存储器映射和堆栈



5.1.1 程序计数器

程序计数器（Program Counter，PC）指定要取出执行的指令地址。PC 为 21 位宽，并且保存在 3 个独立的 8 位寄存器中。其中的低字节称为 PCL 寄存器，该寄存器是可读写的。高字节，即 PCH 寄存器，存储 PC<15:8> 位，不可直接读写。可以通过 PCLATH 寄存器更新 PCH 寄存器。最高字节称为 PCU，该寄存器存储 PC<20:16> 位，它也不能直接读写。通过 PCLATU 寄存器更新 PCU 寄存器。

通过执行写 PCL 的操作，可以将 PCLATH 和 PCLATU 的内容传送到程序计数器。类似的，通过执行读 PCL 的操作，可以将程序计数器的两个高字节传送到 PCLATH 和 PCLATU。这对于计算 PC 的偏移量很有用（见**第 5.1.4.1 节 “计算 GOTO”**）。

PC 在程序存储器中按字节寻址。为防止 PC 不能正确获取指令字，需要将 PCL 的最低有效位固定取值为 0。PC 每次加 2 来连续寻址程序存储器中的指令。

CALL、 RCALL 和 GOTO 等程序转移指令直接将地址写入程序计数器。对于这些指令，PCLATH 和 PCLATU 的内容将不会被传送到程序计数器。

5.1.2 返回地址堆栈

用于存放返回地址的堆栈允许保存最多 31 个程序调用地址和中断向量。当执行 CALL 或 RCALL 指令或者响应中断时， PC 值被压入堆栈。而执行 RETURN、 RETLW 或 RETFIE 指令时， PC 值从堆栈弹出。 PCLATU 和 PCLATH 不受 RETURN 或 CALL 指令的影响。

通过 21 位的 RAM 和 5 位的堆栈指针 （STKPTR）来实现 31 级堆栈操作。堆栈既不占用程序存储空间也不占用数据存储空间。堆栈指针可以读写，并且通过栈顶的特殊文件寄存器可以读写栈顶地址。也可使用这些寄存器将数据压入堆栈，或将数据从堆栈弹出。

执行 CALL 类型指令引起进栈操作：堆栈指针首先加 1，并且将 PC 的内容写入堆栈指针指向的地址单元（PC 已经指向 CALL 的下一条指令）。执行 RETURN 类型指令时，引起出栈操作：STKPTR 寄存器所指向的地址单元的内容被传送给 PC，然后堆栈指针减 1。

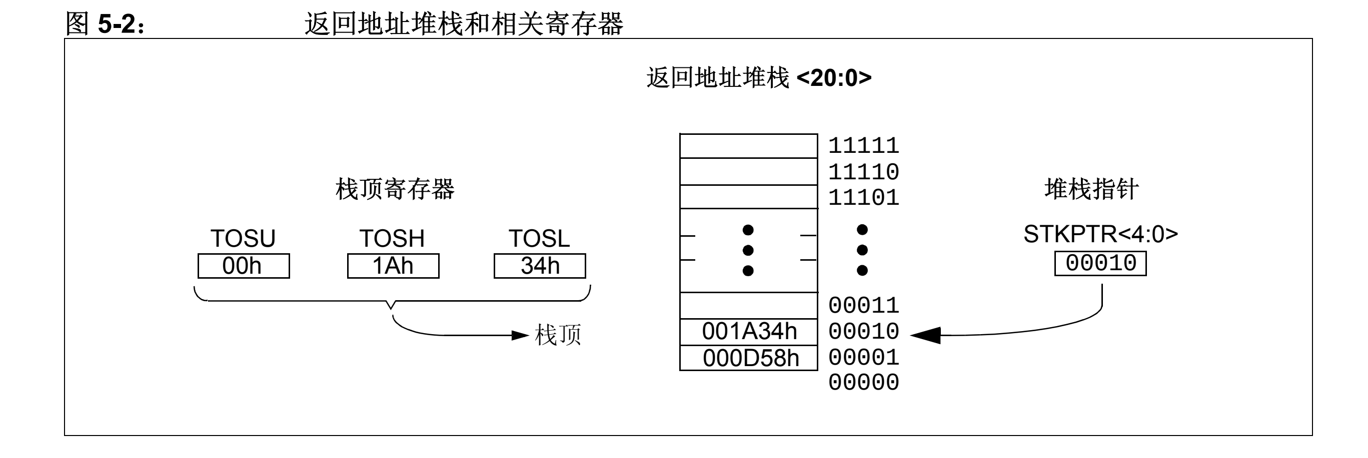
所有复位后，堆栈指针被初始化为 00000。堆栈指针值 00000 不指向任何 RAM 单元，它只是一个复位值。状态位表明堆栈是满、上溢还是下溢。

5.1.2.1 访问栈顶

只有栈顶（Top-of-Stack， TOS）是可读写的。有 3 个寄存器 TOSU:TOSH:TOSL 用于保存 STKPTR 寄存器所指向的堆栈单元的内容（图 5-2）。这可以让用户在必要时实现软件堆栈。在 CALL、 RCALL 或中断后，软件可以通过读取 TOSU:TOSH:TOSL 寄存器来读取进栈值。这些值可以被置入用户定义的软件堆栈。返回时，软件将这些值存回 TOSU:TOSH:TOSL 并执行返回。

为防止对堆栈的意外操作，访问堆栈时用户必须禁止全局中断允许位。

图 5-2： 返回地址堆栈和相关寄存器



 初稿

5.1.2.2 返回堆栈指针 （STKPTR）

STKPTR 寄存器（寄存器 5-1）包含堆栈指针值、STKFUL（堆栈满）状态位和 STKUNF（堆栈下溢）状态位。堆栈指针值可为 0 到 31 之间的整数。向堆栈压入值前，堆栈指针加 1；而从堆栈弹出值后，堆栈指针减 1。复位时，堆栈指针值为零。用户可以读写堆栈指针的值。实时操作系统（Real-Time Operating System，RTOS）可以利用此特性对返回堆栈进行维护。

当向堆栈压入 PC 值 31 次（且没有值从堆栈弹出）后，STKFUL 位就会置 1。通过软件或 POR 使 STKFUL 位清零。

堆栈满时执行的操作由 STVREN（堆栈上溢复位使能）配置位的状态决定（有关器件配置位的介绍，见**第 25.1 节 “配置位”**）。如果 STVREN 位已经置 1 （默认），第 31 次进栈将把 （PC + 2）值压入堆栈，将 STKFUL 位置 1，并复位器件。 STKFUL 位将保持置 1，而堆栈指针将被清零。

如果 STVREN 位被清零，第 31 次进栈时 STKFUL 位会被置 1，堆栈指针则加 1 变为 31。任何其他进栈操作都不会覆盖第31次进栈的值，并且STKPTR将保持为31。

当堆栈弹出次数足够卸空堆栈时，下一次出栈操作会向 PC 返回一个零值，并将 STKUNF 位置 1，而堆栈指针则保持为 0。 STKUNF 位将保持置 1，直到被软件清零或发生 POR。

注：	下溢会导致向 PC 返回一个零值，并使程序指向复位向量，此时可以验证堆栈状态并采取相应的操作。这与复位不同，因为 SFR 的内容不受影响。

5.1.2.3 PUSH 和 POP 指令

由于栈顶是可以读写的，因此将值压入堆栈或从堆栈弹出值而不影响程序的正常执行是非常理想的。PIC18 指令集包括两条指令 PUSH 和 POP，它们允许在软件控制下对 TOS 进行操作。可以通过修改 TOSU、 TOSH 和 TOSL，将数据或返回地址压入堆栈。

PUSH 指令将当前的 PC 值压入堆栈。具体的操作为，先将堆栈指针加 1，再将当前 PC 值装入堆栈。

POP 指令通过将堆栈指针减 1 来丢弃当前的 TOS 值。然后前一个进栈值就成为 TOS 值。

寄存器 5-1： STKPTR：堆栈指针寄存器

R/C-0	R/C-0	U-0	R/W-0	R/W-0	R/W-0	R/W-0	R/W-0
STKFUL[(1)]	STKUNF[(1)]	—	SP4	SP3	SP2	SP1	SP0
bit 7							bit 0

bit 7 **STKFUL：**堆栈满标志位 [(1)]
1 = 堆栈满或上溢
0 = 堆栈未满或未上溢

bit 6 **STKUNF：**堆栈下溢标志位 [(1)]
1 = 发生堆栈下溢
0 = 未发生堆栈下溢

bit 5 未用：读为 0

bit 4-0 **SP4:SP0：**堆栈指针地址位

注 **1：**通过用户软件或 POR 清零 bit 7 和 bit 6。

图注：			
R = 可读位	W = 可写位	U = 未用位	C = 只可清零位
-n = 上电复位时的值	1 = 置 1	0 = 清零	x = 未知

5.1.2.4 堆栈满和下溢复位

通过将配置寄存器 4L 中的 STVREN 位置 1 可以在出现堆栈上溢和堆栈下溢条件时使器件复位。当 STVREN 位置 1 时，堆栈满或堆栈下溢条件会将相应的 STKFUL 或 STKUNF 位置 1，然后使器件复位。当 STVREN 位清零时，堆栈满或堆栈下溢条件会将相应的 STKFUL 或 STKUNF 位置 1，但不会使器件复位。通过用户软件或上电复位使 STKFUL 或 STKUNF 位清零。

5.1.3 快速寄存器堆栈

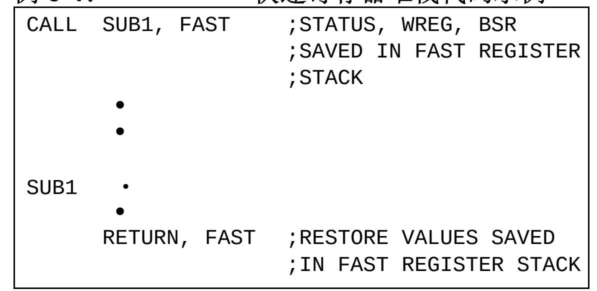
为 Status、WREG 和 BSR 寄存器提供的快速寄存器堆栈具有从中断 “快速返回” 的功能。每个寄存器堆栈的深度仅为 1 级，并且不可读写。当处理器转入中断向量处执行指令时，此堆栈装入对应寄存器的当前值。所有中断源都会将值压入堆栈寄存器。如果使用 RETFIE 和 FAST 指令从中断返回，这些寄存器堆栈中的值会被重新装回对应的寄存器。

如果同时允许了低优先级中断和高优先级中断，从低优先级中断返回时，无法可靠地使用堆栈寄存器。如果在为低优先级中断提供服务时，发生了高优先级中断，则低优先级中断存储在堆栈寄存器中的值将被覆盖。在这些情况下，用户必须在低优先级中断期间用软件保存关键寄存器中的值。

如果未使用中断优先级，所有中断都可以使用快速寄存器堆栈从中断返回。如果没有使用中断，快速寄存器堆栈可以用于在子程序调用结束后恢复 Status、WREG 和 BSR 寄存器。要将快速寄存器堆栈用于子程序调用，必须执行 CALL label 和 FAST 指令将 Status、WREG 和 BSR 寄存器的内容存入快速寄存器堆栈。在调用结束后，执行 RETURN 和 FAST 指令，从快速寄存器堆栈中弹出并恢复这些寄存器的值。

例 5-1给出了一个在子程序调用和返回期间使用快速寄存器堆栈的源代码示例。

例 5-1：　快速寄存器堆栈代码示例



```
CALL  SUB1, FAST    ;STATUS, WREG, BSR
                    ;SAVED IN FAST REGISTER
                    ;STACK
      •
      •

SUB1   •
      •
      RETURN, FAST  ;RESTORE VALUES SAVED
                    ;IN FAST REGISTER STACK
```

5.1.4 程序存储器中的查找表

有些编程需要在程序存储器中创建数据结构或查找表。对于 PIC18 器件，有两种方法可以实现查找表：

- 计算 GOTO
- 表读

5.1.4.1 计算 GOTO

计算 GOTO 是通过向程序计数器加一个偏移量来实现的。例 5-2 给出了一个示例。

使用 ADDWF PCL 指令和一组 RETLW nn 指令可以创建一个查找表。在调用该表前，会先将查找表的偏移量装入 W 寄存器。被调用程序的第一条指令是 ADDWF PCL 指令。接下去执行的是一条 RETLW nn 指令，它将数值 nn 返回给调用函数。

偏移量 （WREG 中的值）指定程序计数器应该增加的字节数，其值应当为 2 的倍数 （LSb = 0）。

若使用这种方法，每个指令单元只能存储一个数据字节，并且要求在返回地址堆栈中还有空闲空间。

例 5-2：　使用偏移量计算 GOTO

```
       MOVF    OFFSET, W
       CALL    TABLE
ORG    nn00h
TABLE  ADDWF   PCL
       RETLW   nnh
       RETLW   nnh
       RETLW   nnh
       .
       .
       .
```

5.1.4.2 表读和表写

有一种更好的方法可以将数据存储在程序存储器中，该方法允许在每个指令单元存储 2 个字节的数据。

使用表读和表写，每个程序字可以存储 2 个字节的查找表数据。表指针寄存器 （TBLPTR）指定字节地址，而表锁存器 （TABLAT）存储从程序存储器读取或写入的数据。每次写入 / 读出程序存储器的数据为一个字节。

在第 **6.1 节 “表读和表写”** 中将进一步讨论表读和表写操作。

5.2 PIC18 指令周期

5.2.1 时钟机制

单片机时钟输入信号，无论来自内部或外部时钟源，都会在器件内部被 4 分频用来产生 4 个不重叠的正交时钟信号，即 Q1、 Q2、 Q3 和 Q4。程序计数器在每个 Q1 递增，并在 Q4 期间从程序存储器取指并将指令锁存到指令寄存器中。指令的译码和执行在下一个 Q1 到 Q4 周期完成。图 5-3 所示为时钟和指令执行的流程图。

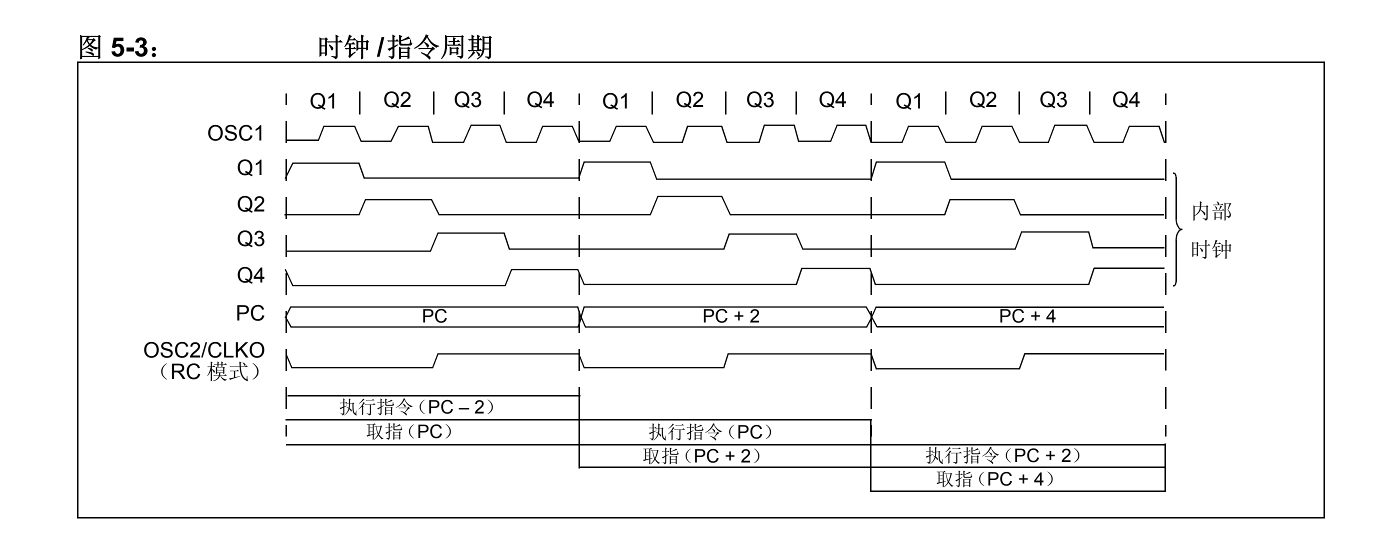
5.2.2 指令流 / 流水线

1 个指令周期由 4 个 Q 周期组成：Q1 到 Q4。指令的取指和执行是以流水线的形式进行的，在一个指令周期进行取指，而在下一个指令周期译码并执行指令。但由于是流水线操作，所以每条指令的等效执行时间都是一个指令周期。如果某条指令改变了程序计数器的值 （如 GOTO 指令），则需要两个指令周期才能完成该指令（见例 5-3）。

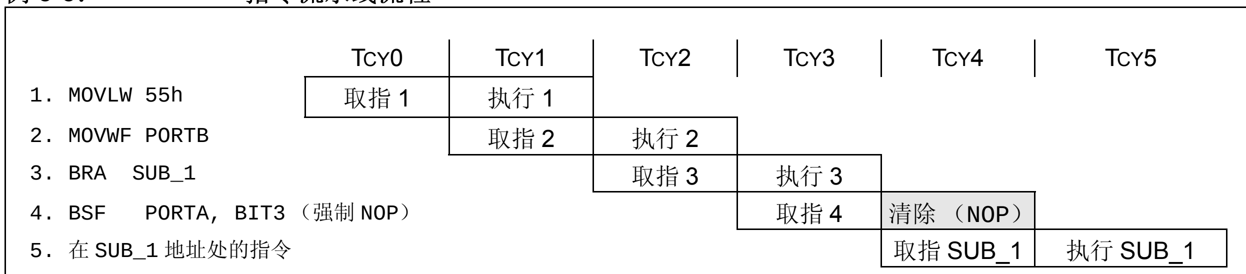
在 Q1 周期，程序计数器 （PC）递增，开始取指。

在指令的执行周期操作如下：在 Q1 周期，将所取指令锁存到指令寄存器 （Instruction Register， IR）。然后在 Q2、Q3 和 Q4 周期中进行指令的译码和执行。其中读数据存储器 （读操作数）发生在 Q2 周期，写操作发生在 Q4 周期 （写目标单元）。

图 5-3： 时钟 / 指令周期



例 5-3： 指令流水线流程



注： 除程序跳转指令之外，所有指令都是单周期指令。由于程序跳转指令将导致一条已取指令从流水线清除，需要重新取指，然后执行指令，所以程序跳转指令需用两个周期。

5.2.3 程序存储器中的指令

程序存储器按字节寻址。指令以 2 字节或 4 字节形式存储在程序存储器中。指令字的最低有效字节始终存储在地址为偶数的程序存储器单元中 （LSb = 0）。要保证正确指向指令单元，PC 必须以 2 为单位递增，并且 LSb 总是读为 0 （见第 **5.1.1 节 “程序计数器”**）。

图 5-4 给出了指令字存储在程序寄存器中的一个示例。

CALL 和 GOTO 指令在指令中嵌入了程序存储器的绝对地址。指令总是存储为一个字长，因而指令所包含的数据为字地址。字地址会写入 PC<20:1>，由 PC 在程序存储器中访问目标地址单元。图 5-4 中的指令 2 给出了指令 GOTO 0006h 在程序存储器中的译码过程。程序跳转指令也采取同样的方式对相对地址偏移量进行译码。在跳转指令中的偏移量代表单字指令数，PC 将以此作为偏移量跳转到指定的地址单元。第 **26.0 节 “指令集综述”** 提供了指令集的更多详情。

图 5-4： 程序存储器中的指令

程序存储器字节单元 →

指令			LSB = 1	LSB = 0	字地址 ↓
					000000h
					000002h
					000004h
					000006h
指令 1：	MOVLW	055h	0Fh	55h	000008h
指令 2：	GOTO	0006h	EFh	03h	00000Ah
			F0h	00h	00000Ch
指令 3：	MOVFF	123h, 456h	C1h	23h	00000Eh
			F4h	56h	000010h
					000012h
					000014h

5.2.4 双字指令

标准的 PIC18 指令集有 4 条双字指令：CALL、MOVFF、GOTO 和 LSFR。在所有情况下，这些指令第二个字的高 4 位总是 1111，而其余 12 位是一个立即数数据，通常为一个数据存储器地址。

指令的高 4 位 1111 代表一条特殊的 NOP 指令。指令执行的正确顺序为：执行完第一个字之后立即按顺序访问并使用第二个字中的数据。如果由于某些原因跳过了第一个字并自行执行指令的第二个字，那么将转而执行一条 NOP 指令。如果双字指令跟在更改 PC 的条件指令后，就有必要执行此操作。例 5-4 说明了其执行过程。

> **注：** 有关扩展指令集中双字指令的信息，请参见第 **5.5 节 “程序存储器和扩展的指令集”**。

例 5-4： 双字指令

情形 1：

目标代码	源代码
0110 0110 0000 0000	TSTFSZ REG1 ; is RAM location 0?
1100 0001 0010 0011	MOVFF REG1, REG2 ; No, skip this word
1111 0100 0101 0110	; Execute this word as a NOP
0010 0100 0000 0000	ADDWF REG3 ; continue code

情形 2：

目标代码	源代码
0110 0110 0000 0000	TSTFSZ REG1 ; is RAM location 0?
1100 0001 0010 0011	MOVFF REG1, REG2 ; Yes, execute this word
1111 0100 0101 0110	; 2nd word of instruction
0010 0100 0000 0000	ADDWF REG3 ; continue code

5.3 数据存储器构成

> 注： 当使能 PIC18 扩展指令集时，数据存储器某些方面的操作会有所改变。如需更多信息，请参见第 **5.6 节 "数据存储器和扩展的指令集"**。

PIC18 器件中的数据存储器是以静态 RAM 的方式实现的。在数据存储器中，每个寄存器有 12 位地址，数据存储容量最多可达 4096 个字节。存储空间被分为 16 个存储区（Bank），每个存储区包含256 个字节； PIC18F2455/2550/4455/4550 器件只有 8个存储区可用，总存储容量为2048个字节。图 5-5 显示了该器件的数据存储器构成。

数据存储器由特殊功能寄存器（SFR）和通用寄存器（General Purpose Register， GPR）组成。 SFR 用于单片机和外设功能模块的控制和状态显示，而 GPR 则用于在用户应用程序中存储数据和高速暂存操作。任何未用单元的读取值均为 0。

此指令集和架构支持对所有存储区的操作。可以通过直接、间接或变址寻址模式访问整个数据存储器。本节后面的部分将讨论寻址模式。

为了确保能在一个周期内存取常用寄存器（SFR 和所选的 GPR），PIC18 器件设置了快速操作存储区（Access Bank）。这是一个 256 字节的存储空间，可实现对 SFR 和 GPR Bank 0 的低地址单元的快速存取，而无需使用 BSR。第 **5.3.3 节 "快速操作存储区"** 提供了对于快速操作 RAM 的详细说明。

5.3.1 USB RAM

数据存储器中 Bank 4 到 Bank 7 实际上被映射到特殊双端口 RAM 中。当禁止 USB 模块时，这些存储区中的 GPR 的操作与普通数据存储空间中的 GPR 相同。

当使能 USB 模块时，这些存储区被分配为用于 USB 操作的缓冲 RAM。这一区域由单片机内核和 USB 串行接口引擎（Serial Interface Engine， SIE）共享，并用于在这两者之间直接传送数据。

理论上可以使用未分配为 USB 缓冲器的 USB RAM 临时存储正常的中间结果或存储其他变量。但事实上，由于缓冲器分配的动态性能，这样做的风险极大。此外，当使能 USB 模块时，Bank 4 用于 USB 缓冲器管理，并在使能模块期间不可用于其他目的。

第 **17.0 节 "通用串行总线（USB）"** 给出了 USB RAM 和缓冲器操作的更多信息。

5.3.2 存储区选择寄存器 （BSR）

存储容量较大的数据存储器需要有效的寻址机制，以便对所有地址进行快速存取。理想状况下，这意味着没有必要为每次读写操作提供整个地址。PIC18 器件是使用 RAM分区机制实现快速存取的。该机制将存储空间分成连续的 16 个 256 字节的存储区。根据不同的指令，可以通过完整的 12 位地址，或通过 8 位低字节地址加 4 位存储区指针来直接寻址每个单元。

PIC18 指令集中的大部分指令都使用存储区指针，该指针被称为存储区选择寄存器（Bank Select Register，BSR）。BSR 保存单元地址的高 4 位，而指令本身则包括单元地址的低 8 位。只使用 BSR 的低 4 位（BSR3:BSR0），其高 4 位未用，始终为 0 且不能被写入。可以通过使用 MOVLB 指令直接装载 BSR。

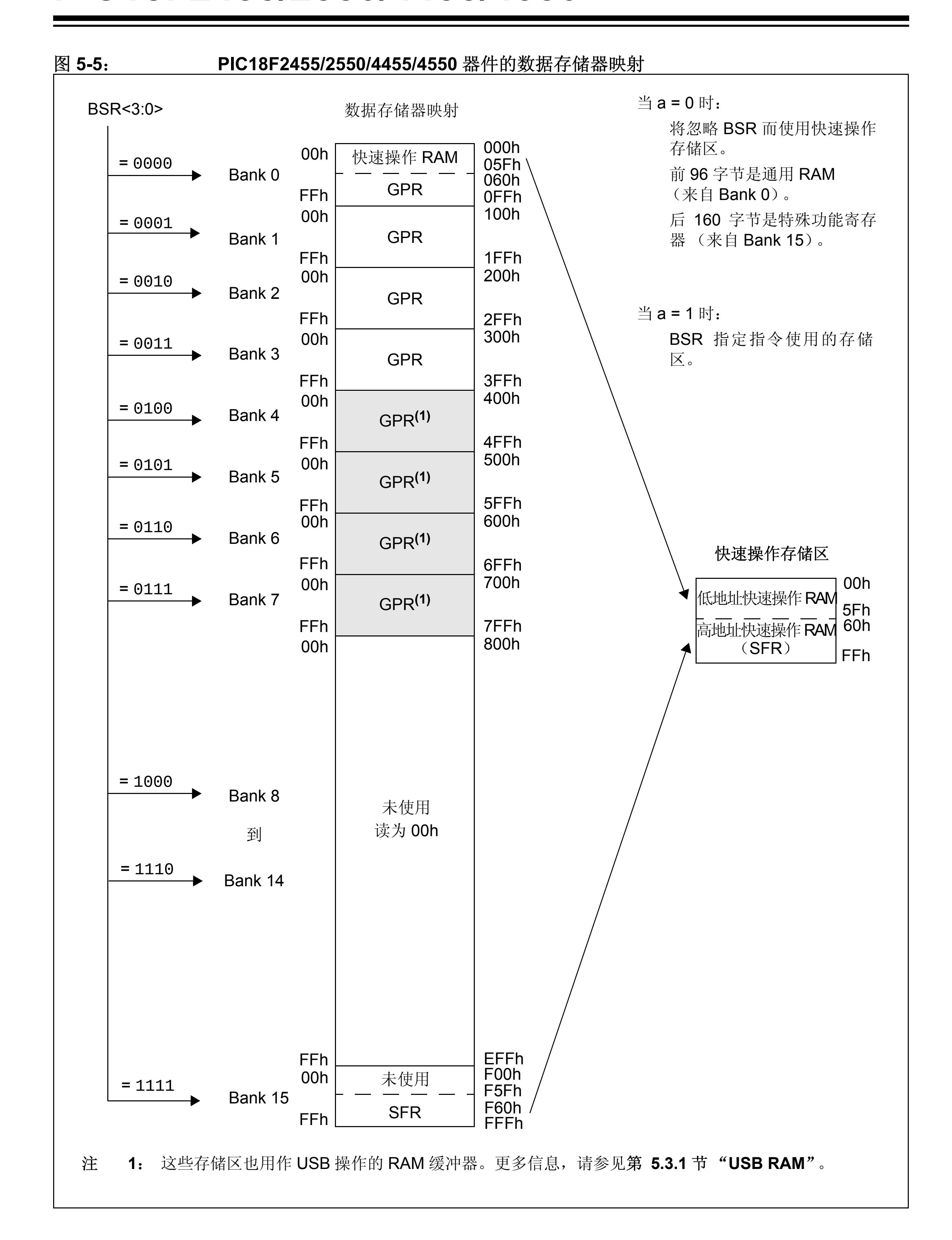
BSR 的值指向数据存储器中的存储区。指令中的 8 位地址指向存储区中的存储单元，可以将它看作是以存储区下边界为起点的偏移量。图 5-6 所示是 BSR 的值与存储区之间的关系。

由于最多有 16 个寄存器共享同一个低位地址，用户必须非常小心以确保在执行数据读或写之前选择了正确的存储区。例如，当 BSR 为 0Fh 时，将程序数据写入地址 F9h 将导致程序计数器复位。

当选择存储区时，只有可用的存储区可以被读写。对未用存储区的写入操作将被忽略，而读未用的存储区会返回 0。虽然是这样， Status 寄存器仍然会受到影响。图 5-5 中的数据存储器映射图指出了可使用的存储区。

在 PIC18 的内核指令集中，只有 MOVFF 指令指定源寄存器和目标寄存器的完整 12 位地址。此指令在执行时完全忽略 BSR 的值。其他指令仅包含作为操作数的低位地址，而且必须使用 BSR 或快速操作存储区来寻址目标寄存器。

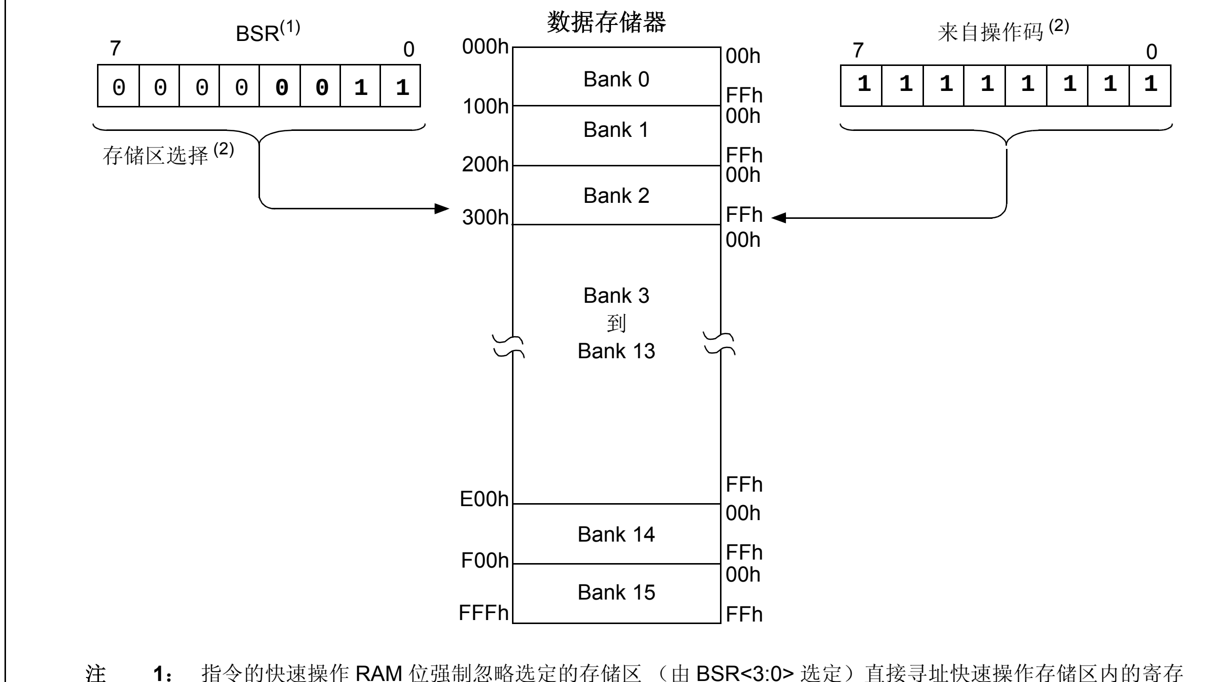
图 5-5： PIC18F2455/2550/4455/4550 器件的数据存储器映射



注 1： 这些存储区也用作 USB 操作的 RAM 缓冲器。更多信息，请参见第 **5.3.1 节 “USB RAM”**。



图 5-6： 使用存储区选择寄存器（直接寻址）



注 1： 指令的快速操作 RAM 位强制忽略选定的存储区（由 BSR<3:0> 选定）直接寻址快速操作存储区内的寄存器。

2： MOVFF 指令嵌入了完整的 12 位地址。

5.3.3 快速操作存储区

使用 BSR 和指令内嵌的 8 位地址可以使用户对整个数据存储空间进行寻址，这同时意味着用户必须始终确保选择了正确的存储区。否则，可能会从错误的单元读取数据或将数据写入错误的单元。如果本来是向 GPR 进行写操作，却将结果写入了 SFR，后果是非常严重的。但是在每次向数据存储器进行读或写操作时，验证和 / 或更改 BSR 可能会降低代码的执行效率。

为了连续访问大多数常用的数据存储单元，必须为数据存储器配置快速操作存储区，这使得用户无需指定 BSR 即可访问被映射的存储区。快速操作存储区由 Bank 0 的前 96 个字节（00h-5Fh）和 Bank 15 的后 160 个字节（60h-FFh）组成。低半部分就是 “快速操作 RAM”，由 GPR 组成。高地址的那一半被映射为器件的 SFR。这两个区域在快速操作存储区中连续映射，并且可以使用 8 位地址进行线性寻址（图 5-5）。

快速操作存储区供包括快速操作 RAM 位（指令中的 a 参数）的 PIC18 内核指令使用。当 a 等于 1 时，指令使用 BSR 和包含在操作码中的 8 位地址来对数据存储器进行寻址。但是当 a 为 0 时，指令被强制使用快速操作存储区地址映射； BSR 的当前值被忽略。

该 “强制” 寻址方式可使指令在一个周期内对数据地址进行操作，而无需首先更新 BSR。这意味着用户可以更有效地对 8 位地址为 60h 及以上的 SFR 进行取值和操作。地址低于 60h 的快速操作 RAM 非常适合于存储那些用户可能需要快速存取的数据值（如直接计算结果或常用程序变量）。快速操作 RAM 还可以实现更快速、代码效率更高的现场保护和变量存取。

当使能扩展的指令集（XINST 配置位 = 1）时，快速操作存储区的映射略有不同。**第 5.6.3 节 “在立即数变址寻址模式下映射快速操作存储区”** 更详细地讨论了此操作。

5.3.4 通用寄存器

PIC18 器件可能在 GRP 区中划分了一部分存储区。这部分存储区为数据 RAM，所有指令都可以访问它。GPR 区从 Bank 0 的底部（地址 000h）开始向上直到 SFR 区的底部。上电复位不会将 GPR 初始化，其他复位也不会改变其内容。

5.3.5 特殊功能寄存器

特殊功能寄存器（Special Function Registers，SFR）是 CPU 和外设模块用来控制器件操作的寄存器。这类寄存器在数据存储空间中以静态 RAM 的形式实现。SFR 起始于数据存储器的顶部（FFFh）并且向下扩展到 Bank 15 的上半部分（从 F60h 到 FFFh）。表 5-1 和表 5-2 列出了这些寄存器。

SFR 可分为两类：一类与“内核”器件功能（ALU、复位和中断）有关，另一类与外设功能有关。在相关的章节中将对复位和中断寄存器进行说明，而本章后面的部分将对 ALU 的 Status 寄存器进行说明。与外设功能部件操作相关的寄存器在外设的章节中进行说明。

SFR 通常位于受其控制的外设中。未使用的 SFR 单元不可用，它们的读取值为 `0`。

表 5-1： PIC18F2455/2550/4455/4550 器件的特殊功能寄存器映射

地址	名称	地址	名称	地址	名称	地址	名称	地址	名称
FFFh	TOSU	FDFh	INDF2[1]	FBFh	CCPR1H	F9Fh	IPR1	F7Fh	UEP15
FFEh	TOSH	FDEh	POSTINC2[1]	FBEh	CCPR1L	F9Eh	PIR1	F7Eh	UEP14
FFDh	TOSL	FDDh	POSTDEC2[1]	FBDh	CCP1CON	F9Dh	PIE1	F7Dh	UEP13
FFCh	STKPTR	FDCh	PREINC2[1]	FBCh	CCPR2H	F9Ch	—[2]	F7Ch	UEP12
FFBh	PCLATU	FDBh	PLUSW2[1]	FBBh	CCPR2L	F9Bh	OSCTUNE	F7Bh	UEP11
FFAh	PCLATH	FDAh	FSR2H	FBAh	CCP2CON	F9Ah	—[2]	F7Ah	UEP10
FF9h	PCL	FD9h	FSR2L	FB9h	—[2]	F99h	—[2]	F79h	UEP9
FF8h	TBLPTRU	FD8h	STATUS	FB8h	BAUDCON	F98h	—[2]	F78h	UEP8
FF7h	TBLPTRH	FD7h	TMR0H	FB7h	ECCP1DEL	F97h	—[2]	F77h	UEP7
FF6h	TBLPTRL	FD6h	TMR0L	FB6h	ECCP1AS	F96h	TRISE[3]	F76h	UEP6
FF5h	TABLAT	FD5h	T0CON	FB5h	CVRCON	F95h	TRISD[3]	F75h	UEP5
FF4h	PRODH	FD4h	—[2]	FB4h	CMCON	F94h	TRISC	F74h	UEP4
FF3h	PRODL	FD3h	OSCCON	FB3h	TMR3H	F93h	TRISB	F73h	UEP3
FF2h	INTCON	FD2h	HLVDCON	FB2h	TMR3L	F92h	TRISA	F72h	UEP2
FF1h	INTCON2	FD1h	WDTCON	FB1h	T3CON	F91h	—[2]	F71h	UEP1
FF0h	INTCON3	FD0h	RCON	FB0h	SPBRGH	F90h	—[2]	F70h	UEP0
FEFh	INDF0[1]	FCFh	TMR1H	FAFh	SPBRG	F8Fh	— 2)	F6Fh	UCFG
FEEh	POSTINC0[1]	FCEh	TMR1L	FAEh	RCREG	F8Eh	—[2]	F6Eh	UADDR
FEDh	POSTDEC0[1]	FCDh	T1CON	FADh	TXREG	F8Dh	LATE[3]	F6Dh	UCON
FECh	PREINC0[1]	FCCh	TMR2	FACh	TXSTA	F8Ch	LATD[3]	F6Ch	USTAT
FEBh	PLUSW0[1]	FCBh	PR2	FABh	RCSTA	F8Bh	LATC	F6Bh	UEIE
FEAh	FSR0H	FCAh	T2CON	FAAh	—[2]	F8Ah	LATB	F6Ah	UEIR
FE9h	FSR0L	FC9h	SSPBUF	FA9h	EEADR	F89h	LATA	F69h	UIE
FE8h	Ω REG	FC8h	SSPADD	FA8h	EEDATA	F88h	—[2]	F68h	UIR
FE7h	INDF1[1]	FC7h	SSPSTAT	FA7h	EECON2[1]	F87h	—[2]	F67h	UFRMH
FE6h	POSTINC1[1]	FC6h	SSPCON1	FA6h	EECON1	F86h	—[2]	F66h	UFRML
FE5h	POSTDEC1[1]	FC5h	SSPCON2	FA5h	—[2]	F85h	—[2]	F65h	SPPCON[3]
FE4h	PREINC1[1]	FC4h	ADRESH	FA4h	—[2]	F84h	PORTE	F64h	SPPEPS[3]
FE3h	PLUSW1[1]	FC3h	ADRESL	FA3h	—[2]	F83h	PORTD[3]	F63h	SPPCFG[3]
FE2h	FSR1H	FC2h	ADCON0	FA2h	IPR2	F82h	PORTC	F62h	SPPDATA[3]
FE1h	FSR1L	FC1h	ADCON1	FA1h	PIR2	F81h	PORTB	F61h	—[2]
FE0h	BSR	FC0h	ADCON2	FA0h	PIE2	F80h	PORTA	F60h	—[2]

注 1：非物理寄存器。
2：不存在的寄存器，读为 0。
3：这些寄存器仅在 40/44 引脚器件上存在。

 初稿

表 5-2： 文件寄存器汇总 （**PIC18F2455/2550/4455/4550**）

寄存器名称	Bit 7	Bit 6	Bit 5	Bit 4	Bit 3	Bit 2	Bit 1	Bit 0	POR 与 BOR 时的值	详情请见：（页）
TOSU	—	—	—	栈顶最高字节 （TOS<20:16>）					---0 0000	51，58
TOSH	栈顶高字节 （TOS<15:8>）								0000 0000	51，58
TOSL	栈顶低字节 （TOS<7:0>）								0000 0000	51，58
STKPTR	STKFUL	STKUNF	—	返回堆栈指针					00-0 0000	51，59
PCLATU	—	—	—	PC<20:16> 的保持寄存器					---0 0000	51，58
PCLATH	PC<15:8> 的保持寄存器								0000 0000	51，58
PCL	PC 低字节 （PC<7:0>）								0000 0000	51，58
TBLPTRU	—	—	bit 21[1]	程序存储器表指针最高字节 （TBLPTR<20:16>）					--00 0000	51，82
TBLPTRH	程序存储器表指针高字节 （TBLPTR<15:8>）								0000 0000	51，82
TBLPTRL	程序存储器表指针低字节 （TBLPTR<7:0>）								0000 0000	51，82
TABLAT	程序存储器表锁存器								0000 0000	51，82
PRODH	乘积寄存器高字节								xxxx xxxx	51，95
PRODL	乘积寄存器低字节								xxxx xxxx	51，95
INTCON	GIE/GIEH	PEIE/GIEL	TMR0IE	INT0IE	RBIE	TMR0IF	INT0IF	RBIF	0000 000x	51，99
INTCON2	RBPU	INTEDG0	INTEDG1	INTEDG2	—	TMR0IP	—	RBIP	1111 -1-1	51，100
INTCON3	INT2IP	INT1IP	—	INT2IE	INT1IE	—	INT2IF	INT1IF	11-0 0-00	51，101
INDF0	使用 FSR0 的内容来寻址数据存储器——FSR0 的值不变 （不是物理寄存器）								N/A	51，73
POSTINC0	使用 FSR0 的内容来寻址数据存储器——FSR0 的值后增 （不是物理寄存器）								N/A	51，74
POSTDEC0	使用 FSR0 的内容来寻址数据存储器——FSR0 的值后减 （不是物理寄存器）								N/A	51，74
PREINC0	使用 FSR0 的内容来寻址数据存储器——FSR0 的值预增 （不是物理寄存器）								N/A	51，74
PLUSW0	使用 FSR0 的内容来寻址数据存储器——FSR0 的值预增 （不是物理寄存器）， FSR0 的偏移量由 W 寄存器提供								N/A	51，74
FSR0H	—	—	—	—	间接数据存储器地址指针 0 的高字节				---- 0000	51，73
FSR0L	间接数据存储器地址指针 0 的低字节								xxxx xxxx	51，73
Ω REG	工作寄存器								xxxx xxxx	51
INDF1	使用 FSR1 的内容来寻址数据存储器——FSR1 的值不变 （不是物理寄存器）								N/A	51，73
POSTINC1	使用 FSR1 的内容来寻址数据存储器——FSR1 的值后增 （不是物理寄存器）								N/A	51，74
POSTDEC1	使用 FSR1 的内容来寻址数据存储器——FSR1 的值后减 （不是物理寄存器）								N/A	51，74
PREINC1	使用 FSR1 的内容来寻址数据存储器——FSR1 的值预增 （不是物理寄存器）								N/A	51，74
PLUSW1	使用 FSR1 的内容来寻址数据存储器——FSR1 的值预增 （不是物理寄存器）， FSR1 的偏移量由 W 寄存器提供								N/A	51，74
FSR1H	—	—	—	—	间接数据存储器地址指针 1 的高字节				---- 0000	51，73
FSR1L	间接数据存储器地址指针 1 的低字节								xxxx xxxx	51，73
BSR	—	—	—	—	存储区选择寄存器				---- 0000	52，63
INDF2	使用 FSR2 的内容来寻址数据存储器——FSR2 的值不变 （不是物理寄存器）								N/A	52，73
POSTINC2	使用 FSR2 的内容来寻址数据存储器——FSR2 的值后增 （不是物理寄存器）								N/A	52，74
POSTDEC2	使用 FSR2 的内容来寻址数据存储器——FSR2 的值后减 （不是物理寄存器）								N/A	52，74
PREINC2	使用 FSR2 的内容来寻址数据存储器——FSR2 的值预增 （不是物理寄存器）								N/A	52，74
PLUSW2	使用 FSR2 的内容来寻址数据存储器——FSR2 的值预增 （不是物理寄存器）， FSR2 的偏移量由 W 寄存器提供								N/A	52，74
FSR2H	—	—	—	—	间接数据存储器地址指针 2 的高字节				---- 0000	52，73
FSR2L	间接数据存储器地址指针 2 的低字节								xxxx xxxx	52，73
STATUS	—	—	—	N	OV	Z	DC	C	---x xxxx	52，71
TMR0H	Timer0 寄存器的高字节								0000 0000	52，127
TMR0L	Timer0 寄存器的低字节								xxxx xxxx	52，127
T0CON	TMR0ON	T08BIT	T0CS	T0SE	PSA	T0PS2	T0PS1	T0PS0	1111 1111	52，125

图注： x = 未知， u = 不变， — = 未用， q = 取值视情况而定

注
- **1：** TBLPTRU 中的 bit 21 允许访问器件配置位。
- **2：** 仅当 BOREN<1:0> = 01 时， SBOREN 位才可用，否则该位读为 0。
- **3：** 这些寄存器和 / 或位在 28 引脚器件上不存在并且读为 0。此处给出了 40/44 引脚器件的复位值。未用位以 “—”表示。
- **4：** 根据不同的主振荡模式， RA6 被配置为端口引脚。当禁止端口引脚时，所有相关位均读为 0。
- **5：** 仅当 MCLRE 配置位清零时， RE3 才可用作端口引脚；否则该位读为 0。
- **6：** 只有在禁止 USB 模块 （UCON<3> = 0）时， RC4 和 RC5 才可用作端口引脚。

表 5-2：　　文件寄存器汇总 （PIC18F2455/2550/4455/4550） （续）

寄存器名称	Bit 7	Bit 6	Bit 5	Bit 4	Bit 3	Bit 2	Bit 1	Bit 0	POR 与 BOR 时的值	详情请见：（页）
OSCCON	IDLEN	IRCF2	IRCF1	IRCF0	OSTS	IOFS	SCS1	SCS0	0100 q000	52，32
HLVDCON	VDIRMAG	—	IRVST	HLVDEN	HLVDL3	HLVDL2	HLVDL1	HLVDL0	0-00 0101	52，273
WDTCON	—	—	—	—	—	—	—	SWDTEN	--- ---0	52，292
RCON	IPEN	SBOREN[2]	—	$\overline{RI}$	$\overline{TO}$	$\overline{PD}$	$\overline{POR}$	$\overline{BOR}$	0q-1 11q0	52，44
TMR1H	Timer1 寄存器的高字节								xxxx xxxx	52，133
TMR1L	Timer1 寄存器的低字节								xxxx xxxx	52，133
T1CON	RD16	T1RUN	T1CKPS1	T1CKPS0	T1OSCEN	$\overline{T1SYNC}$	TMR1CS	TMR1ON	0000 0000	52，129
TMR2	Timer2 寄存器								0000 0000	52，136
PR2	Timer2 周期寄存器								1111 1111	52，136
T2CON	—	T2OUTPS3	T2OUTPS2	T2OUTPS1	T2OUTPS0	TMR2ON	T2CKPS1	T2CKPS0	-000 0000	52，135
SSPBUF	SSP 接收缓冲器 / 发送寄存器								xxxx xxxx	52，194，202
SSPADD	I^2C™ 从动模式下的 SSP 地址寄存器。 I^2C™ 主控模式下的 SSP 波特率重载寄存器。								0000 0000	52，202
SSPSTAT	SMP	CKE	D/$\overline{A}$	P	S	R/$\overline{W}$	UA	BF	0000 0000	52，194，203
SSPCON1	WCOL	SSPOV	SSPEN	CKP	SSPM3	SSPM2	SSPM1	SSPM0	0000 0000	52，195，204
SSPCON2	GCEN	ACKSTAT	ACKDT	ACKEN	RCEN	PEN	RSEN	SEN	0000 0000	52，205
ADRESH	A/D 结果寄存器的高字节								xxxx xxxx	52，262
ADRESL	A/D 结果寄存器的低字节								xxxx xxxx	52，262
ADCON0	—	—	CHS3	CHS2	CHS1	CHS0	GO/$\overline{DONE}$	ADON	--00 0000	52，253
ADCON1	—	—	VCFG1	VCFG0	PCFG3	PCFG2	PCFG1	PCFG0	--00 0qqq	52，254
ADCON2	ADFM	—	ACQT2	ACQT1	ACQT0	ADCS2	ADCS1	ADCS0	0-00 0000	52，255
CCPR1H	捕捉 / 比较 /PWM 寄存器 1 的高字节								xxxx xxxx	53，142
CCPR1L	捕捉 / 比较 /PWM 寄存器 1 的低字节								xxxx xxxx	53，142
CCP1CON	P1M1[3]	P1M0[3]	DC1B1	DC1B0	CCP1M3	CCP1M2	CCP1M1	CCP1M0	0000 0000	53，141，149
CCPR2H	捕捉 / 比较 /PWM 寄存器 2 的高字节								xxxx xxxx	53，142
CCPR2L	捕捉 / 比较 /PWM 寄存器 2 的低字节								xxxx xxxx	53，142
CCP2CON	—	—	DC2B1	DC2B0	CCP2M3	CCP2M2	CCP2M1	CCP2M0	--00 0000	53，141
BAUDCON	ABDOVF	RCIDL	—	SCKP	BRG16	—	WUE	ABDEN	01-0 0-00	53，236
ECCP1DEL	PRSEN	PDC6[3]	PDC5[3]	PDC4[3]	PDC3[3]	PDC2[3]	PDC1[3]	PDC0[3]	0000 0000	53，158
ECCP1AS	ECCPASE	ECCPAS2	ECCPAS1	ECCPAS0	PSSAC1	PSSAC0	PSSBD1[3]	PSSBD0[3]	0000 0000	53，159
CVRCON	CVREN	CVROE	CVRR	CVRSS	CVR3	CVR2	CVR1	CVR0	0000 0000	53，269
CMCON	C2OUT	C1OUT	C2INV	C1INV	CIS	CM2	CM1	CM0	0000 0111	53，263
TMR3H	Timer3 寄存器的高字节								xxxx xxxx	53，139
TMR3L	Timer3 寄存器的低字节								xxxx xxxx	53，139
T3CON	RD16	T3CCP2	T3CKPS1	T3CKPS0	T3CCP1	$\overline{T3SYNC}$	TMR3CS	TMR3ON	0000 0000	53，137
SPBRGH	EUSART 波特率发生器寄存器的高字节								0000 0000	53，237
SPBRG	EUSART 波特率发生器寄存器的低字节								0000 0000	53，237
RCREG	EUSART 接收寄存器								0000 0000	53，244
TXREG	EUSART 发送寄存器								0000 0000	53，242
TXSTA	CSRC	TX9	TXEN	SYNC	SENDB	BRGH	TRMT	TX9D	0000 0010	53，234
RCSTA	SPEN	RX9	SREN	CREN	ADDEN	FERR	OERR	RX9D	0000 000x	53，235

图注： x = 未知， u = 不变， — = 未用， q = 取值视情况而定

注 1： TBLPTRU 中的 bit 21 允许访问器件配置位。

2： 仅当 BOREN<1:0> = 01 时， SBOREN 位才可用，否则该位读为 0。

3： 这些寄存器和 / 或位在 28 引脚器件上不存在并且读为 0。此处给出了 40/44 引脚器件的复位值。未用位以 “—” 表示。

4： 根据不同的主振荡模式， RA6 被配置为端口引脚。当禁止端口引脚时，所有相关位均读为 0。

5： 仅当 MCLRE 配置位清零时， RE3 才可用作端口引脚；否则该位读为 0。

6： 只有在禁止 USB 模块 （UCON<3> = 0）时， RC4 和 RC5 才可用作端口引脚。

表 **5-2**： 文件寄存器汇总 （**PIC18F2455/2550/4455/4550**） （续）

寄存器名称	Bit 7	Bit 6	Bit 5	Bit 4	Bit 3	Bit 2	Bit 1	Bit 0	POR 与 BOR 时的值	详情请见：（页）
EEADR	EEPROM 地址寄存器								0000 0000	53，89
EEDATA	EEPROM 数据寄存器								0000 0000	53，89
EECON2	EEPROM 控制寄存器 2 （非物理寄存器）								0000 0000	53，80
EECON1	EEPGD	CFGS	—	FREE	WRERR	WREN	WR	RD	xx-0 x000	53，81
IPR2	OSCFIP	CMIP	USBIP	EEIP	BCLIP	HLVDIP	TMR3IP	CCP2IP	1111 1111	54，107
PIR2	OSCFIF	CMIF	USBIF	EEIF	BCLIF	HLVDIF	TMR3IF	CCP2IF	0000 0000	54，103
PIE2	OSCFIE	CMIE	USBIE	EEIE	BCLIE	HLVDIE	TMR3IE	CCP2IE	0000 0000	54，105
IPR1	SPPIP(3)	ADIP	RCIP	TXIP	SSPIP	CCP1IP	TMR2IP	TMR1IP	1111 1111	54，106
PIR1	SPPIF(3)	ADIF	RCIF	TXIF	SSPIF	CCP1IF	TMR2IF	TMR1IF	0000 0000	54，102
PIE1	SPPIE(3)	ADIE	RCIE	TXIE	SSPIE	CCP1IE	TMR2IE	TMR1IE	0000 0000	54，104
OSCTUNE	INTSRC	—	—	TUN4	TUN3	TUN2	TUN1	TUN0	0--0 0000	54，28
TRISE(3)	—	—	—	—	—	TRISE2	TRISE1	TRISE0	---- -111	54，124
TRISD(3)	TRISD7	TRISD6	TRISD5	TRISD4	TRISD3	TRISD2	TRISD1	TRISD0	1111 1111	54，122
TRISC	TRISC7	TRISC6	—	—	—	TRISC2	TRISC1	TRISC0	11-- -111	54，119
TRISB	TRISB7	TRISB6	TRISB5	TRISB4	TRISB3	TRISB2	TRISB1	TRISB0	1111 1111	54，116
TRISA	—	TRISA6(4)	TRISA5	TRISA4	TRISA3	TRISA2	TRISA1	TRISA0	-111 1111	54，113
LATE(3)	—	—	—	—	—	LATE2	LATE1	LATE0	---- -xxx	54，124
LATD(3)	LATD7	LATD6	LATD5	LATD4	LATD3	LATD2	LATD1	LATD0	xxxx xxxx	54，122
LATC	LATC7	LATC6	—	—	—	LATC2	LATC1	LATC0	---- -xxx	54，119
LATB	LATB7	LATB6	LATB5	LATB4	LATB3	LATB2	LATB1	LATB0	xxxx xxxx	54，116
LATA	—	LATA6(4)	LATA5	LATA4	LATA3	LATA2	LATA1	LATA0	-xxx xxxx	54，113
PORTE	RDPU(3)	—	—	—	RE3(5)	RE2(3)	RE1(3)	RE0(3)	0--- x000	54，123
PORTD(3)	RD7	RD6	RD5	RD4	RD3	RD2	RD1	RD0	xxxx xxxx	54，122
PORTC	RC7	RC6	RC5(6)	RC4(6)	—	RC2	RC1	RC0	xxxx -xxx	54，119
PORTB	RB7	RB6	RB5	RB4	RB3	RB2	RB1	RB0	xxxx xxxx	54，116
PORTA	—	RA6(4)	RA5	RA4	RA3	RA2	RA1	RA0	-x0x 0000	54，113
UEP15	—	—	—	EPHSHK	EPCONDIS	EPOUTEN	EPINEN	EPSTALL	---0 0000	55，169
UEP14	—	—	—	EPHSHK	EPCONDIS	EPOUTEN	EPINEN	EPSTALL	---0 0000	55，169
UEP13	—	—	—	EPHSHK	EPCONDIS	EPOUTEN	EPINEN	EPSTALL	---0 0000	55，169
UEP12	—	—	—	EPHSHK	EPCONDIS	EPOUTEN	EPINEN	EPSTALL	---0 0000	55，169
UEP11	—	—	—	EPHSHK	EPCONDIS	EPOUTEN	EPINEN	EPSTALL	---0 0000	55，169
UEP10	—	—	—	EPHSHK	EPCONDIS	EPOUTEN	EPINEN	EPSTALL	---0 0000	55，169
UEP9	—	—	—	EPHSHK	EPCONDIS	EPOUTEN	EPINEN	EPSTALL	---0 0000	55，169
UEP8	—	—	—	EPHSHK	EPCONDIS	EPOUTEN	EPINEN	EPSTALL	---0 0000	55，169
UEP7	—	—	—	EPHSHK	EPCONDIS	EPOUTEN	EPINEN	EPSTALL	---0 0000	55，169
UEP6	—	—	—	EPHSHK	EPCONDIS	EPOUTEN	EPINEN	EPSTALL	---0 0000	55，169
UEP5	—	—	—	EPHSHK	EPCONDIS	EPOUTEN	EPINEN	EPSTALL	---0 0000	55，169
UEP4	—	—	—	EPHSHK	EPCONDIS	EPOUTEN	EPINEN	EPSTALL	---0 0000	55，169
UEP3	—	—	—	EPHSHK	EPCONDIS	EPOUTEN	EPINEN	EPSTALL	---0 0000	55，169
UEP2	—	—	—	EPHSHK	EPCONDIS	EPOUTEN	EPINEN	EPSTALL	---0 0000	55，169
UEP1	—	—	—	EPHSHK	EPCONDIS	EPOUTEN	EPINEN	EPSTALL	---0 0000	55，169
UEP0	—	—	—	EPHSHK	EPCONDIS	EPOUTEN	EPINEN	EPSTALL	---0 0000	55，169

图注： x = 未知， u = 不变， — = 未用， q = 取值视情况而定

注 **1：** TBLPTRU 中的 bit 21 允许访问器件配置位。

2： 仅当 BOREN<1:0> = 01 时， SBOREN 位才可用，否则该位读为 0。

3： 这些寄存器和 / 或位在 28 引脚器件上不存在并且读为 0。此处给出了 40/44 引脚器件的复位值。未用位以 “—” 表示。

4： 根据不同的主振荡模式， RA6 被配置为端口引脚。当禁止端口引脚时，所有相关位均读为 0。

5： 仅当 MCLRE 配置位清零时， RE3 才可用作端口引脚；否则该位读为 0。

6： 只有在禁止 USB 模块 （UCON<3> = 0）时， RC4 和 RC5 才可用作端口引脚。

表 5-2： 文件寄存器汇总 （PIC18F2455/2550/4455/4550） （续）

寄存器名称	Bit 7	Bit 6	Bit 5	Bit 4	Bit 3	Bit 2	Bit 1	Bit 0	POR 与 BOR 时的值	详情请见：（页）
UCFG	UTEYE	UOEMON	—	UPUEN	UTRDIS	FSEN	PPB1	PPB0	00-0 0000	55，166
UADDR	—	ADDR6	ADDR5	ADDR4	ADDR3	ADDR2	ADDR1	ADDR0	-000 0000	55，170
UCON	—	PPBRST	SE0	PKTDIS	USBEN	RESUME	SUSPND	—	-0x0 000-	55，164
USTAT	—	ENDP3	ENDP2	ENDP1	ENDP0	DIR	PPBI	—	-xxx xxx-	55，168
UEIE	BTSEE	—	—	BTOEE	DFN8EE	CRC16EE	CRC5EE	PIDEE	0--0 0000	55，181
UEIR	BTSEF	—	—	BTOEF	DFN8EF	CRC16EF	CRC5EF	PIDEF	0--0 0000	55，180
UIE	—	SOFIE	STALLIE	IDLEIE	TRNIE	ACTVIE	UERRIE	URSTIE	-000 0000	55，179
UIR	—	SOFIF	STALLIF	IDLEIF	TRNIF	ACTVIF	UERRIF	URSTIF	-000 0000	55，178
UFRMH	—	—	—	—	—	FRM10	FRM9	FRM8	---- -xxx	55，170
UFRML	FRM7	FRM6	FRM5	FRM4	FRM3	FRM2	FRM1	FRM0	xxxx xxxx	55，170
SPPCON(3)	—	—	—	—	—	—	SPPOWN	SPPEN	---- --00	55，187
SPPEPS(3)	RDSPP	WRSPP	—	SSPBUSY	ADDR3	ADDR2	ADDR1	ADDR0	00-0 0000	55，191
SPPCFG(3)	CLKCFG1	CLKCFG0	CSEN	CLK1EN	WS3	WS2	WS1	WS0	0000 0000	55，188
SPPDATA(3)	DATA7	DATA6	DATA5	DATA4	DATA3	DATA2	DATA1	DATA0	0000 0000	55，192

图注： x = 未知， u = 不变， — = 未用， q = 取值视情况而定

注 1： TBLPTRU 中的 bit 21 允许访问器件配置位。

2： 仅当 BOREN<1:0> = 01 时， SBOREN 位才可用，否则该位读为 0。

3： 这些寄存器和 / 或位在 28 引脚器件上不存在并且读为 0。此处给出了 40/44 引脚器件的复位值。未用位以 “—” 表示。

4： 根据不同的主振荡模式， RA6 被配置为端口引脚。当禁止端口引脚时，所有相关位均读为 0。

5： 仅当 MCLRE 配置位清零时， RE3 才可用作端口引脚；否则该位读为 0。

6： 只有在禁止 USB 模块 （UCON<3> = 0）时， RC4 和 RC5 才可用作端口引脚。

 初稿

5.3.6 STATUS 寄存器

Status 寄存器，如寄存器 5-2 所示，包含 ALU 的算术运算状态。和其他 SFR 一样，它可以是任何指令的操作数。

如果一条影响 Z、DC、C、OV 或 N 位的指令以 Status 寄存器作为目标寄存器，指令执行的结果将不会被直接写入，而是根据执行结果来更新寄存器状态位。所以，当执行一条把 Status 寄存器作为目标寄存器的指令后，Status 寄存器的结果可能和预想的不一样。例如，CLRF STATUS 指令的执行结果是，将 Z 位置 1 而其他状态位保持不变（000u u1uu）。

因此，建议仅使用 BCF、BSF、SWAPF、MOVFF 和 MOVWF 指令来改变 Status 寄存器，因为这些指令不会影响该寄存器中的 Z、 C、 DC、 OV 或 N 位。

欲知其他不会影响状态位的指令，请参见表 26-2 和表 26-3 中的指令集汇总。

> **注：** 在减法运算中，C 和 DC 位分别作为借位和辅助借位标志位。

寄存器 5-2： STATUS 寄存器

U-0	U-0	U-0	R/W-x	R/W-x	R/W-x	R/W-x	R/W-x
—	—	—	N	OV	Z	DC	C
bit 7							bit 0

bit 7-5 **未用：**读为 0

bit 4 **N：**负标志位
此位用于有符号的算术运算（通过 2 进制补码方式进行）。它表明结果是否为负（ALU MSB = 1）。
1 = 结果为负
0 = 结果为正

bit 3 **OV：**溢出位
此位用于有符号的算术运算（通过 2 进制补码方式进行）。表明运算结果溢出了 7 位二进制数的范围，溢出将导致符号位（bit 7）发生改变。
1 = 有符号的算术运算中发生溢出（本次运算）
0 = 没有发生溢出

bit 2 **Z：**全零标志位
1 = 算术运算或逻辑运算结果为零
0 = 算术运算或逻辑运算结果不为零

bit 1 **DC：**辅助进位 / 借位标志位
用于 ADDWF、 ADDLW、 SUBLW 和 SUBWF 指令：
1 = 结果的第 4 个低位发生了进位
0 = 结果的第 4 个低位未发生进位

> **注：** 对于借位，极性是相反的。减法是通过加上第二个操作数的 2 进制补码来实现的。对于移位指令（RRF 和 RLF），此位的值来自源寄存器的 bit 4 或 bit 3。

bit 0 **C：**进位 / 借位标志位
用于 ADDWF、 ADDLW、 SUBLW 和 SUBWF 指令：
1 = 结果的最高有效位发生了进位
0 = 结果的最高有效位未发生进位

> **注：** 对于借位，极性是相反的。减法是通过加上第二个操作数的 2 进制补码来实现的。对于移位指令（RRF 和 RLF），此位的值来自源寄存器的最高位或最低位。

图注：			
R = 可读位	W = 可写位	U = 未用位，读为 0	
-n = 上电复位时的值	1 = 置 1	0 = 清零	x = 未知

5.4 数据寻址模式

注：	当使能 PIC18 扩展指令集时，PIC18 内核指令集中的某些指令的执行会发生改变。更多信息，请参见**第 5.6 节 "数据存储器和扩展的指令集"**。

虽然只能用一种方法（即通过程序计数器）对程序存储器进行寻址，但是可以用多种方法来对数据存储空间进行寻址。大多数指令的寻址模式都是固定的。其他指令最多可以使用 3 种模式，这取决于使用哪些操作数以及是否使能了扩展指令集。

寻址模式有：

- 固有寻址
- 立即数寻址
- 直接寻址
- 间接寻址

当使能扩展指令集（XINST 配置位 = 1）时，还可以使用另一种寻址模式，即立即数变址寻址模式。**第 5.6.1 节 "使用立即数偏移量进行变址寻址"**将更详细地讨论它的操作。

5.4.1 固有和立即数寻址

很多 PIC18 控制指令根本不需要任何参数，执行这些指令要么对整个器件造成影响，要么仅针对一个寄存器进行操作。这种寻址模式就是固有寻址。例如 SLEEP、RESET 和 DAW 指令。

其他指令的工作方式与此类似，但需要操作码中有直接的参数。由于需要一些立即数作为参数，这种寻址模式被称为立即数寻址模式。例如 ADDLW 和 MOVLW，它们分别将立即数值加到或移入 W 寄存器。其他的立即数寻址指令，例如 CALL 和 GOTO，包含一个 20 位的程序存储器地址。

5.4.2 直接寻址

直接寻址在操作码中指定操作的全部或部分源地址和 / 或目标地址。这些选项由指令附带的参数指定。

在 PIC18 内核指令集中，面向位和字节的指令默认情况下使用直接寻址。所有这些指令都包含某个 8 位的直接地址作为它们的最低有效字节。此地址指定作为指令数据源的数据 RAM 的某个寄存器地址（**第 5.3.4 节 "通用寄存器"**）或快速操作存储区（**第 5.3.3 节 "快速操作存储区"**）中的单元地址。

快速操作 RAM 位 a 决定地址的解析方式。当 a 为 1 时，BSR（**第 5.3.2 节 "存储区选择寄存器（BSR）"**）的内容和地址一起用于确定寄存器完整的 12 位地址。当 a 为 0 时，此地址将被解释为快速操作存储区中的一个寄存器。使用快速操作 RAM 寻址有时候也被称为直接强制寻址模式。

有几个指令，比如 MOVFF，在操作码中包含完整的 12 位地址（源地址或目标地址）。在这些情况下，BSR 完全被忽略。

操作结果的目标寄存器由目标位 d 确定。当 d 为 1 时，结果被存储到源寄存器并覆盖它原来的内容。当 d 为 0 时，结果被存回 W 寄存器中。不带 d 参数的指令的目标地址是隐含的，它们是操作的目标寄存器或 W 寄存器。

5.4.3 间接寻址

间接寻址允许用户访问数据存储器中的单元而不需要在指令中给出一个固定的地址。这是通过使用文件选择寄存器（File Select Register， FSR）作为指针指向被读取或写入的单元实现的。由于 FSR 本身作为特殊功能寄存器位于 RAM 中，所以也可以在程序控制下直接对它们进行操作。这使得 FSR 对于在数据存储器中实现诸如表和数组等数据结构非常有用。

也可以使用间接指针操作数（Indirect File Operand，INDF）对寄存器进行间接寻址。这种操作允许自动递增、递减或偏移指针，从而自动控制指针的值。它通过循环提高代码执行效率，如例 5-5所示的清零整个RAM 存储区的操作。

例 5-5： 使用间接寻址清零 RAM（BANK 1）的方法

```
        LFSR   FSR0, 100h ;
NEXT    CLRF   POSTINC0   ; Clear INDF
                          ; register then
                          ; inc pointer
        BTFSS  FSR0H, 1   ; All done with
                          ; Bank1?
        BRA    NEXT       ; NO, clear next
CONTINUE                  ; YES, continue
```

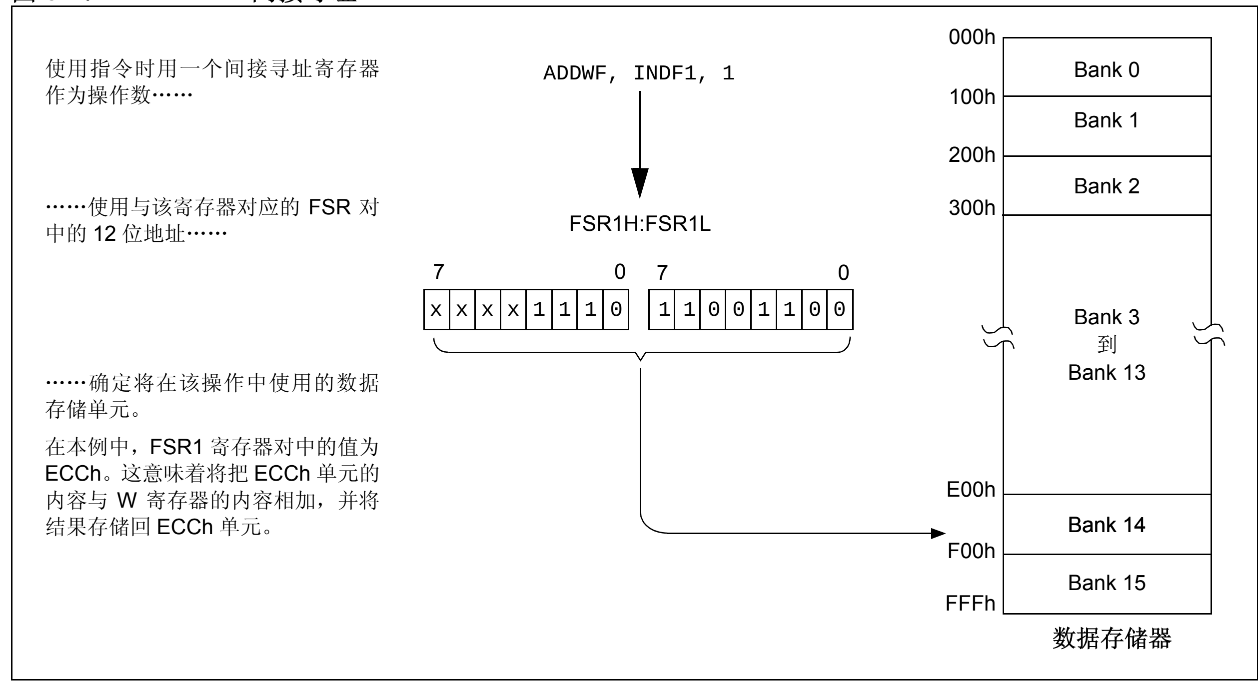

5.4.3.1 FSR 寄存器和 INDF 操作数

间接寻址的核心是三组寄存器：FSR0、FSR1和FSR2。每组寄存器都包含一对8位寄存器：FSRnH和FSRnL。FSRnH 寄存器的高四位未使用，所以每对 FSR 只保存一个 12 位二进制数，从而可以线性寻址整个数据存储空间。因此，FSR 寄存器对被用作数据存储器的地址指针。

间接寻址是通过一组间接指针操作数（从 INDF0 到 INDF2）完成的。这些操作数可以被看作“虚拟”寄存器；它们被映射到SFR空间中而不是通过物理方式实现的。对特定的 INDF 寄存器执行读或写操作实际上访问的是相应的 FSR 寄存器对。例如，读 INDF1 就是读 FSR1H:FSR1L 指向的地址单元中的数据。使用 INDF 寄存器作为操作数的指令实际上使用的是相应的FSR的内容，该内容为指向目标地址的指针。 INDF 操作数只是使用指针的一种较方便的方法。

由于间接寻址使用完整的 12 位地址，因此没有必要进行数据 RAM 分区。因此， BSR 和快速操作 RAM 位的当前内容对于确定目标地址没有影响。

图 5-7： 间接寻址



5.4.3.2 FSR 寄存器和 POSTINC、POSTDEC、PREINC 以及 PLUSW

除了 INDF 操作数之外，每对 FSR 寄存器还有四个额外的间接操作数。和 INDF 一样，它们也是不能直接读写的 “虚拟” 寄存器。访问这些寄存器其实就是访问对应的 FSR 寄存器对，并对其存储的数据进行特定的操作。这些寄存器是：

- POSTDEC：访问 FSR 值，然后自动将它减 1
- POSTINC：访问 FSR 值，然后自动将它加 1
- PREINC：将 FSR 的值加 1，然后在操作中使用该值
- PLUSW：将 W 寄存器中带符号的值 （从 -127 到 128）与 FSR 寄存器中的值相加，并在操作中使用得到的新值。

如前所述，访问 INDF 寄存器使用的是 FSR 寄存器中的值，但不更改此值。同样，访问 PLUSW 寄存器是将 W 寄存器中的值作为 FSR 值的偏移量，该操作不会改变这两个寄存器中的值。访问其他虚拟寄存器会更改 FSR 寄存器的值。

用 POSTDEC、 POSTINC 和 PREINC 对 FSR 进行操作会影响整个寄存器对，也就是说一旦 FSRnL 寄存器从 FFh 到 00h 溢出，会向 FSRnH 寄存器进位。但这些操作的结果不会更改 Status 寄存器中的标志位（如 Z、N 和 OV 等）。

PLUSW 寄存器可以用于在数据存储空间实现变址寻址。通过控制 W 寄存器中的值，用户可以访问相对当前指针地址有固定偏移量的地址单元。在某些应用中，该功能可以被用于在数据存储器内部实现某些非常有用的程序控制结构，如软件堆栈。

5.4.3.3 通过 FSR 对其他 FSR 进行操作

在某些特殊情况下，间接寻址操作以其他 FSR 或虚拟寄存器作为寻址目标。例如，使用 FSR 指向一个虚拟寄存器会导致操作不成功。假设如下特殊情况：FSR0H:FSR0L 保存的是 INDF1 的地址 FE7h。尝试使用 INDF0 作为操作数读取 INDF1 的值，将返回 00h。尝试使用 INDF0 作为操作数写入 INDF1，将会导致执行一条 NOP。

另一方面，使用虚拟寄存器对 FSR 寄存器进行写操作可能会产生与预期不同的结果。在这些情形下，值将被写入 FSR 寄存器对，但 FSR 中的值不会有递增或递减。因此，写入 INDF2 或 POSTDEC2 时会把同样的值写入 FSR2H:FSR2L。

由于 FSR 是在 SFR 空间中映射的物理寄存器，所以可以通过直接寻址对它们进行操作。用户在使用这些寄存器时应该特别小心，尤其是在代码使用间接寻址的情况下。

同样，通常允许通过间接寻址对所有其他的 SFR 进行操作。用户在进行此类操作时应该特别小心，以免更改设置从而影响器件操作。

5.5 程序存储器和扩展的指令集

使用扩展的指令集对程序存储器的操作不产生影响。

使能扩展的指令集将在现有的PIC18指令集的基础上添加 8 条额外的双字指令：ADDFSR、ADDULNK、CALLW、MOVSF、MOVSS、PUSHL、SUBFSR 和 SUBULNK。第 **5.2.4** 节 **“双字指令”**给出了这些指令的执行方式。

5.6 数据存储器和扩展的指令集

使能 PIC18 扩展指令集（XINST 配置位 = 1）将显著地改变数据存储器及其寻址的方式，特别是许多 PIC18 内核指令使用快速操作存储区的方式会有所不同。这是由于扩展的指令集引入了对数据存储空间的新的寻址模式。此模式还会改变使用 FSR2 及其相关操作数进行间接寻址的方式。

同样需要了解哪些部分保持不变。数据存储空间的大小及其线性寻址方式都不会改变。SFR 映射也保持不变。PIC18 内核指令也仍然以直接和间接寻址模式进行操作；固有和立即数指令操作照旧。FSR0 和 FSR1 的间接寻址方式也保持不变。

5.6.1 使用立即数偏移量进行变址寻址

使能 PIC18 扩展的指令集将更改使用 FSR2 寄存器对进行间接寻址的方式。在某些条件下，使用快速操作存储区的指令（即面向位和字节的指令）可以利用指令中的偏移量来执行变址寻址。这种特定的寻址模式被称为使用立即数偏移量的变址寻址或立即数变址寻址模式。

在使用扩展的指令集时，这种寻址模式有如下要求：

- 强制使用快速操作存储区 （a = 0）；且
- 指针地址参数要小于或等于 5Fh。

在这些条件下，指令中的地址不被解析为地址的低字节（与直接寻址中的 BSR 一起使用）或快速操作存储区中的 8 位地址。相反，该值被解析为由 FSR2 指定的地址指针的偏移量。该偏移量和 FSR2 的内容相加以获得操作的目标地址。

5.6.2 受立即数变址寻址模式影响的指令

任何使用直接寻址的 PIC18 内核指令均会受到立即数变址寻址模式的潜在影响，包括所有面向字节和位的指令，或者标准 PIC18 指令集中几乎一半的指令。只有使用固有或立即数寻址模式的指令不受影响。

此外，如果面向字节和位的指令不使用快速操作存储区（快速操作 RAM 位为 1）或包含 60h 以上的地址，它们也不受影响。符合这些条件的指令会像以前一样执行。图 5-8 给出了当使能了扩展的指令集时，各种寻址模式之间的对比。

那些想要在立即数变址寻址模式中使用面向字节或位的指令的用户，应该注意此模式下汇编语法的改变。第 **26.2.1** 节 **“扩展指令的语法”**中将对此进行更详细的说明。

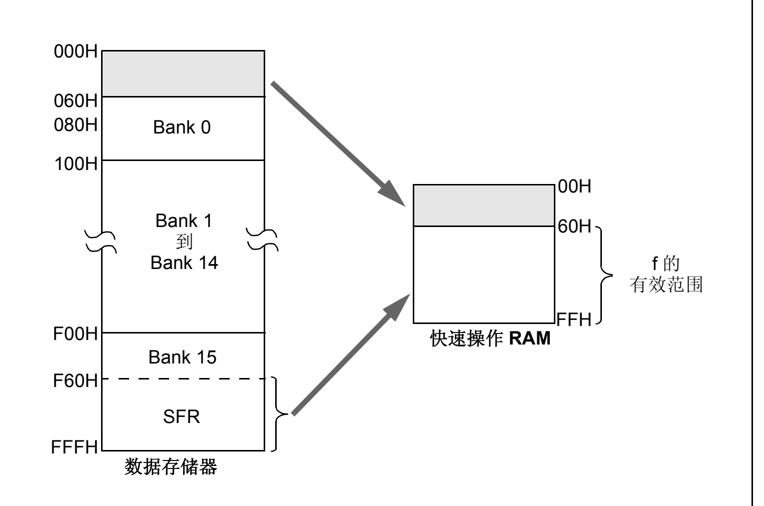
图 5-8： 面向位和字节的指令的寻址方式对比 （使能了扩展的指令集）

示例指令： `ADDWF f, d,a` （操作码： `0010 01da ffff ffff`）

当 a = 0 且 f ≥ 60h：

该指令以直接强制模式执行。f 被解析为快速操作 RAM 中 060h 和 0FFh 之间的单元地址。这实际上是 SFR 或数据存储器中从 F60h 到 0FFh 的存储单元 （Bank 15）。

不可以使用此模式寻址地址低于 60h 的单元。



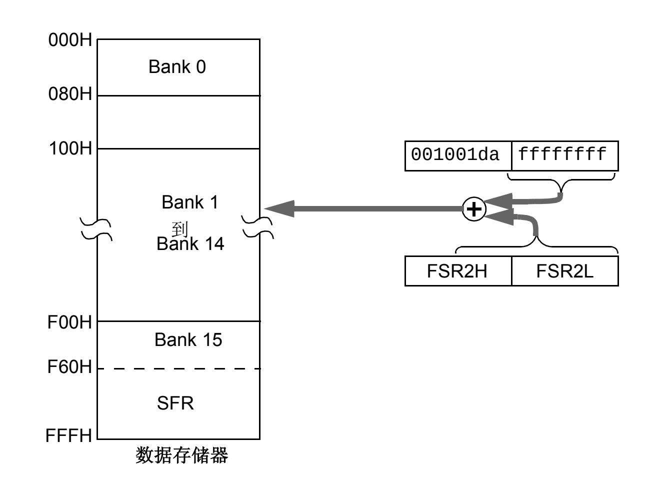
当 a = 0 且 f ≤ 5Fh：

该指令以立即数变址寻址模式执行。f 被解析为 FSR2 中地址值的偏移量。这两个值相加以获得指令的目标寄存器的地址。此地址可以在数据存储空间的任何位置。

注意在此模式中，正确的语法是：

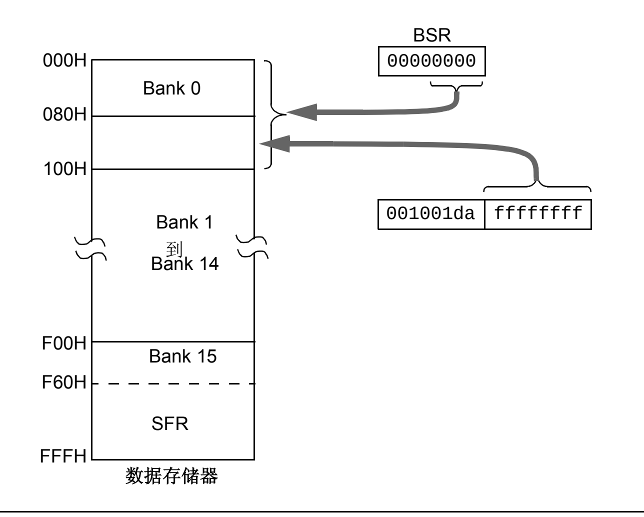
`ADDWF [k], d`

其中 k 就是 f。



当 a = 1 时（f 的所有值）：

指令以直接寻址模式执行 （也被称为直接长地址寻址模式）。f 被解析为数据存储空间的16个存储区中一个单元的地址。存储区由存储区选择寄存器 （BSR）指定。此地址可以在数据存储空间的任何位置。



初稿

5.6.3 在立即数变址寻址模式下映射快速操作存储区

使用立即数变址寻址模式能改变快速操作 RAM 低半部分 （00h 到 5Fh）的映射方式。此模式映射 Bank 0 的内容和由用户定义的可以位于数据存储空间中任何地方的“窗口”内容，而不是仅仅包含 Bank 0 底部的内容。FSR2 的值定义映射到窗口的地址的下边界，而上边界则由 FSR2 加 95 （5Fh）决定。地址为 5Fh 以上的快速操作 RAM 的映射方式如前所述（见第 **5.3.3** 节 **“快速操作存储区”**）。图 5-9显示了在此寻址模式中重新映射的快速操作存储区示例。

快速操作存储区的重新映射*仅适用*于立即数变址寻址模式。使用 BSR （快速操作 RAM 位为 1）的操作和前面一样继续使用直接寻址模式。任何仅使用间接指针操作数（包括 FR2）进行的间接或变址操作都将继续以标准的间接寻址模式进行操作。任何使用快速操作存储区，但包括大于 05Fh 的寄存器地址的指令仍将使用直接寻址和常规的快速操作存储区映射。

5.6.4 立即数变址寻址模式中的 BSR

虽然当使能扩展指令集时会重新映射快速操作存储区，但 BSR 的操作将保持不变。使用 BSR 来选择数据存储区的直接寻址模式，其操作方式与以前描述的相同。

图 5-9： 使用立即数变址寻址重新映射快速操作存储区

示例指令：

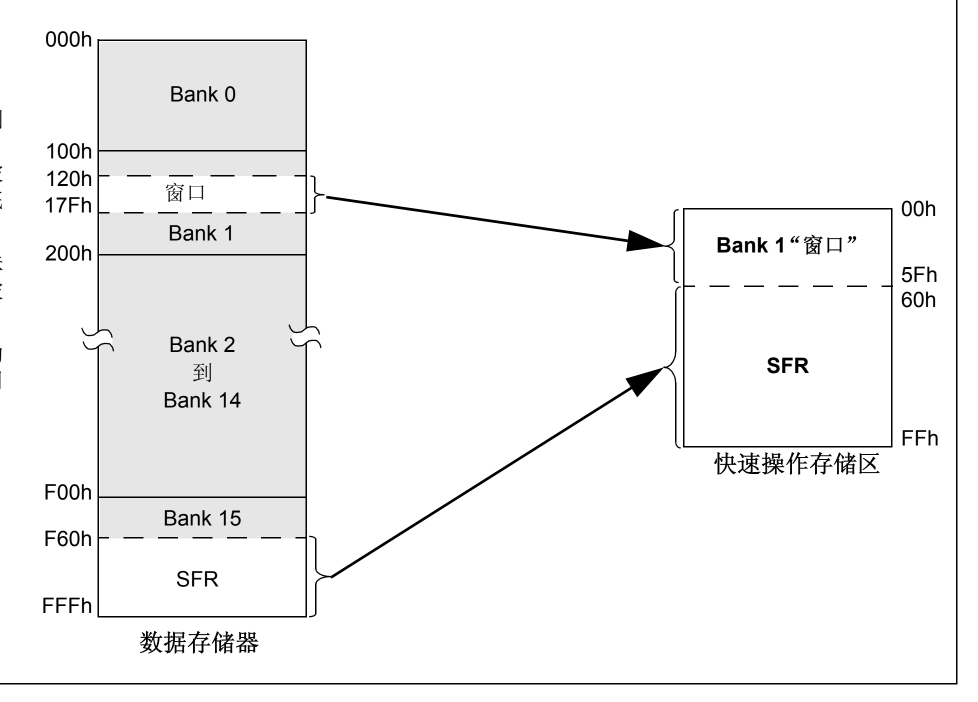
```
ADDWF f, d, a
```

FSR2H:FSR2L = 120h

由 FSR2 指针 （120h）到该指针加上 05Fh （17Fh）指定区域内的存储单元被映射到快速操作 RAM 的底部 （000h 到 05Fh）。

位于 F60h 到 FFFh 的特殊文件寄存器像往常一样被映射到 60h 到 FFh。

在此模式下地址低于5Fh的 Bank 0 不可用。仍然使用 BSR 对它们进行寻址。



注：

6.0 闪存程序存储器

在整个 VDD 范围内，闪存程序存储器在正常工作状态下都是可读写并可擦除的。

对程序存储器执行读操作时每次读取一个字节。对程序存储器执行写操作时每次写入一个32字节的数据块。对程序存储器执行擦除操作每次擦除一个 64 字节的数据块。不允许用户代码执行批量擦除操作。

写或擦除程序存储器将中止取指操作，直到写或擦除操作完成为止。在写或擦除期间不能访问程序存储器，因此无法执行代码。内部编程定时器可终止程序存储器的写入和擦除操作。

写入程序存储器的值不必是有效指令。执行存有无效指令的程序存储器单元会导致执行 NOP 指令。

6.1 表读和表写

为了读取和写入程序存储器，有两种操作可以让处理器在程序存储空间和数据 RAM 之间传送字节：

- 表读 （TBLRD）
- 表写 （TBLWT）

程序存储空间为 16 位宽，而数据 RAM 空间为 8 位宽。表读和表写操作通过一个 8 位寄存器 （TABLAT）在这两个存储空间之间传送数据。

表读操作从程序存储器获取数据并将其存入数据 RAM 空间。图 6-1 显示了程序存储器和数据 RAM 之间的表读操作。

表写操作将数据存储空间中的数据存入程序存储器中的保持寄存器。第 **6.5 节 "写入闪存程序存储器"**将详细介绍将保持寄存器中的内容写入程序存储器的过程。图 6-2 显示了程序存储器和数据 RAM 之间的表写操作。

表操作以字节为单位进行。一个包含数据而非程序指令的表块不必以字为单位。因此，表块可在任何字节地址处开始和结束。如果使用表写操作将可执行代码写入程序存储器，程序指令就需要以字为单位。

图 **6-1**： 表读操作

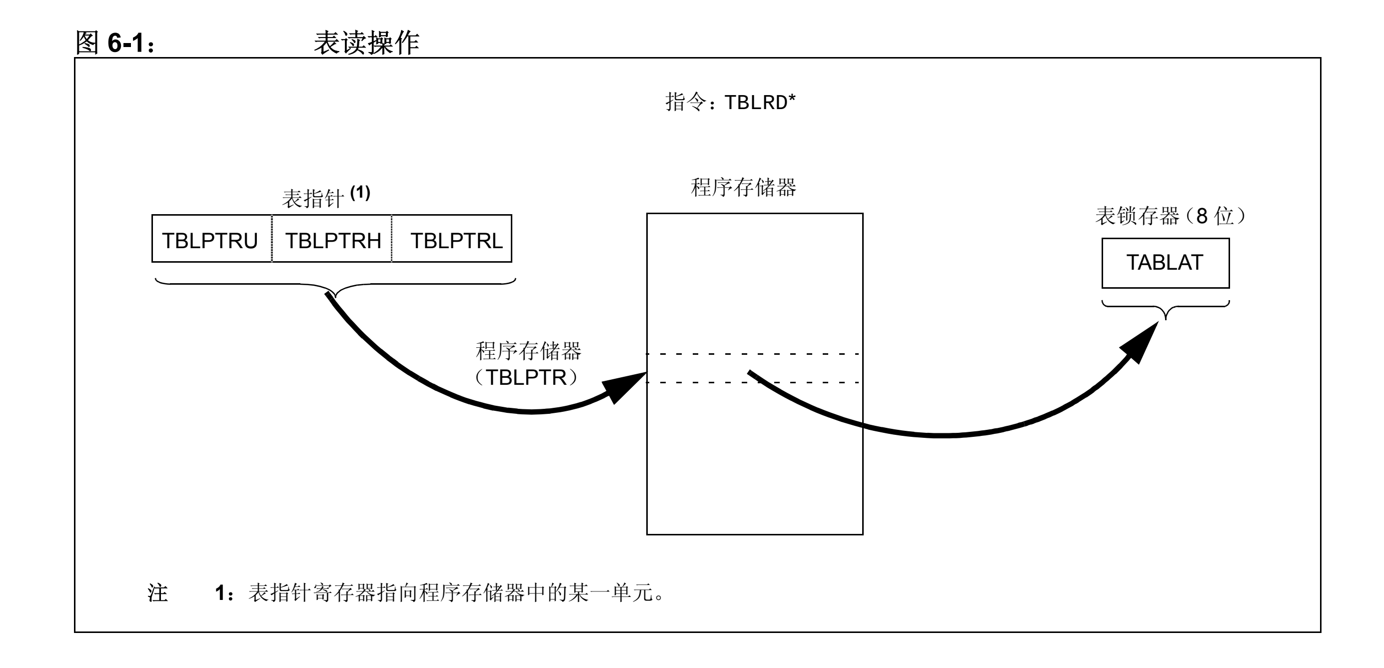
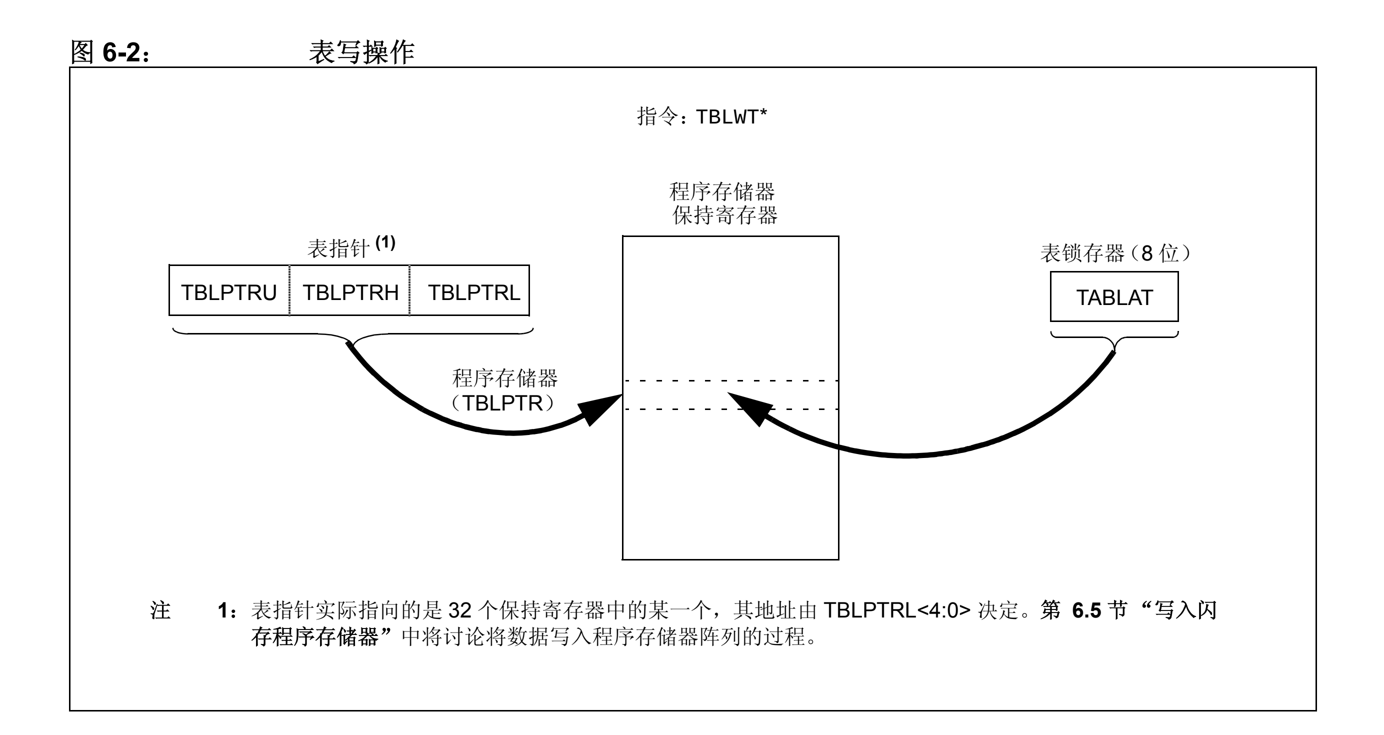


图 **6-2**： 表写操作



注 **1**：表指针实际指向的是 32 个保持寄存器中的某一个，其地址由 TBLPTRL<4:0> 决定。第 **6.5** 节 **"写入闪存程序存储器"**中将讨论将数据写入程序存储器阵列的过程。

6.2 控制寄存器

TBLRD 和 TBLWT 指令要用到几个控制寄存器。其中包括：

- EECON1 寄存器
- EECON2 寄存器
- TABLAT 寄存器
- TBLPTR 寄存器

6.2.1 EECON1 和 EECON2 寄存器

EECON1 寄存器（寄存器 6-1）是访问存储器的控制寄存器。EECON2 寄存器不是物理寄存器，它专用于存储器的擦写操作。读取 EECON2 得到的是全 0。

控制位 EEPGD 决定将访问程序存储器还是数据 EEPROM 存储器。若清零该位，任何后续操作均将针对数据 EEPROM 存储器进行。若该位置 1，任何后续操作均将针对程序存储器进行。

控制位CFGS决定将访问配置寄存器还是程序存储器或数据 EEPROM 存储器。若该位置 1，则无论 EEPGD 如何设置，后续操作都将针对配置寄存器进行（见第 **25.0** 节 **"CPU 的特殊性能"**）。若该位清零，由 EEPGD 决定要访问的存储器。

若将FREE位置1，则允许对程序存储器进行擦除操作。擦除操作由下一条 WR 命令触发。当 FREE 位清零时，则只使能写操作。

若将 WREN 位置 1，则允许写操作。上电时将清零 WREN 位。在 WREN 位置 1 时，WRERR 位将被硬件置 1；当内部编程定时器超时并且写操作完成时，清零 WRERR 位。

注：	如果在正常工作期间，WRERR 的读取值为 1，则表明写操作因复位而提早终止或进行了非法的写操作。

控制位 WR 用于启动写操作。此位只能用软件置 1 而不能清零。写操作完成后，由硬件将其清零。

注：	当写操作完成时 EEIF 中断标志位（PIR2<4>）置 1。此标志位必须用软件清零。

寄存器 6-1： EECON1：数据 EEPROM 控制寄存器 1

R/W-x	R/W-x	U-0	R/W-0	R/W-x	R/W-0	R/S-0	R/S-0
EEPGD	CFGS	—	FREE	WRERR	WREN	WR	RD
bit 7							bit 0

bit 7 **EEPGD**：闪存程序存储器或数据 EEPROM 存储器选择位
1 = 访问闪存程序存储器
0 = 访问数据 EEPROM 存储器

bit 6 **CFGS**：闪存程序存储器 / 数据 EEPROM 存储器或配置寄存器选择位
1 = 访问配置寄存器
0 = 访问闪存程序存储器或数据 EEPROM 存储器

bit 5 未用：读为 0

bit 4 **FREE**：闪存行擦除使能位
1 = 在执行下一条 WR 命令时擦除由 TBLPTR 指定的程序存储器行（在完成擦除操作时清零）
0 = 仅执行写操作

bit 3 **WRERR**：闪存程序 / 数据 EEPROM 错误标志位
1 = 写操作提早终止 （由于正常工作中自定时编程期间的任何复位或非法写入）
0 = 写操作完成

注： WRERR 置 1 时，不会清零 EEPGD 和 CFGS 位。这允许跟踪错误状态。

bit 2 **WREN**：闪存程序 / 数据 EEPROM 写使能位
1 = 允许闪存程序 / 数据 EEPROM 的写周期
0 = 禁止闪存程序 / 数据 EEPROM 的写周期

bit 1 **WR**：写控制位
1 = 启动数据 EEPROM 擦写周期或程序存储器擦写周期
（该操作是自定时的，一旦写入完成该位即由硬件清零。WR 位只能由软件置 1 而不能被其清零。）
0 = EEPROM 的写入周期完成

bit 0 **RD**：读控制位
1 = 启动 EEPROM 读操作（读取需要一个周期。RD 位由硬件清零。软件只能将 RD 位置 1 而不能清零。当 EEPGD = 1 或 CFGS = 1 时， RD 位不能置 1。）
0 = 未启动 EEPROM 读操作

图注：			
R = 可读位	W = 可写位	U = 未用位，读为 0	
-n = 上电复位时的值	1 = 置 1	0 = 清零	x = 未知

6.2.2 表锁存寄存器 （TABLAT）

表锁存器（TABLAT）是映射到 SFR 空间的一个 8 位寄存器。表锁存器用于在程序存储器和数据 RAM 之间传输数据时保存 8 位数据。

6.2.3 表指针寄存器 （TBLPTR）

表指针 （TBLPTR）寄存器在程序存储器中寻址字节。TBLPTR 由 3 个 SFR 寄存器组成：表指针最高字节、表指针高字节和表指针低字节（TBLPTRU:TBLPTRH:TBLPTRL）。这 3 个寄存器合起来组成一个 22 位宽的指针。其中低 21 位可使器件寻址至多 2 MB 的程序存储空间。第 22 位则允许访问器件 ID、用户 ID 以及配置位。

TBLRD 和 TBLWT 指令使用表指针寄存器 TBLPTR。这些指令可以利用 4 种表操作方法中的一种更新 TBLPTR。表 6-1 列出了这些操作。这些操作只会影响 TBLPTR 的低 21 位。

6.2.4 表指针边界

TBLPTR 用于闪存程序存储器的读取、写入和擦除。

当执行 TBLRD 时，TBLPTR 的所有 22 位决定将程序存储器哪个单元的数据读入 TABLAT。

当执行 TBLWT 时，表指针寄存器的低 5 位（TBLPTR<4:0>）决定要写入 32 个程序存储器保持寄存器的哪一个。当开始定时写入程序存储器时 （通过 WR 位），表指针寄存器的高 17 位 （TBLPTR<21:5>）决定要写入哪一个 32 字节的程序存储块。如需更多详情，请参见**第 6.5 节 “写入闪存程序存储器”**。

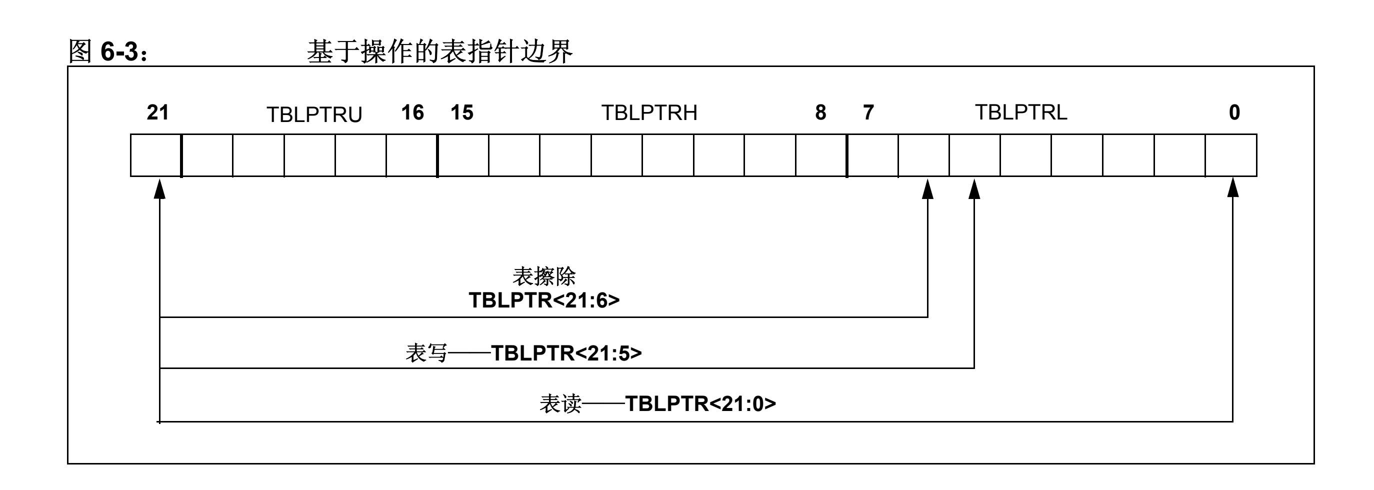
当擦除程序存储器时，表指针的高 16 位（TBLPTR<21:6>）指向将要被擦除的 64 字节块。低 6 位（TBLPTR<5:0>）被忽略。

图 6-3 说明了基于闪存程序存储器操作的 TBLPTR 边界。

表 **6-1**： 使用 **TBLRD** 和 **TBLWT** 指令对表指针操作

示例	表指针操作
TBLRD* TBLWT*	不修改 TBLPTR
TBLRD*+ TBLWT*+	TBLPTR 在读 / 写后递增
TBLRD*- TBLWT*-	TBLPTR 在读 / 写后递减
TBLRD+* TBLWT+*	TBLPTR 在读 / 写前递增

图 **6-3**： 基于操作的表指针边界



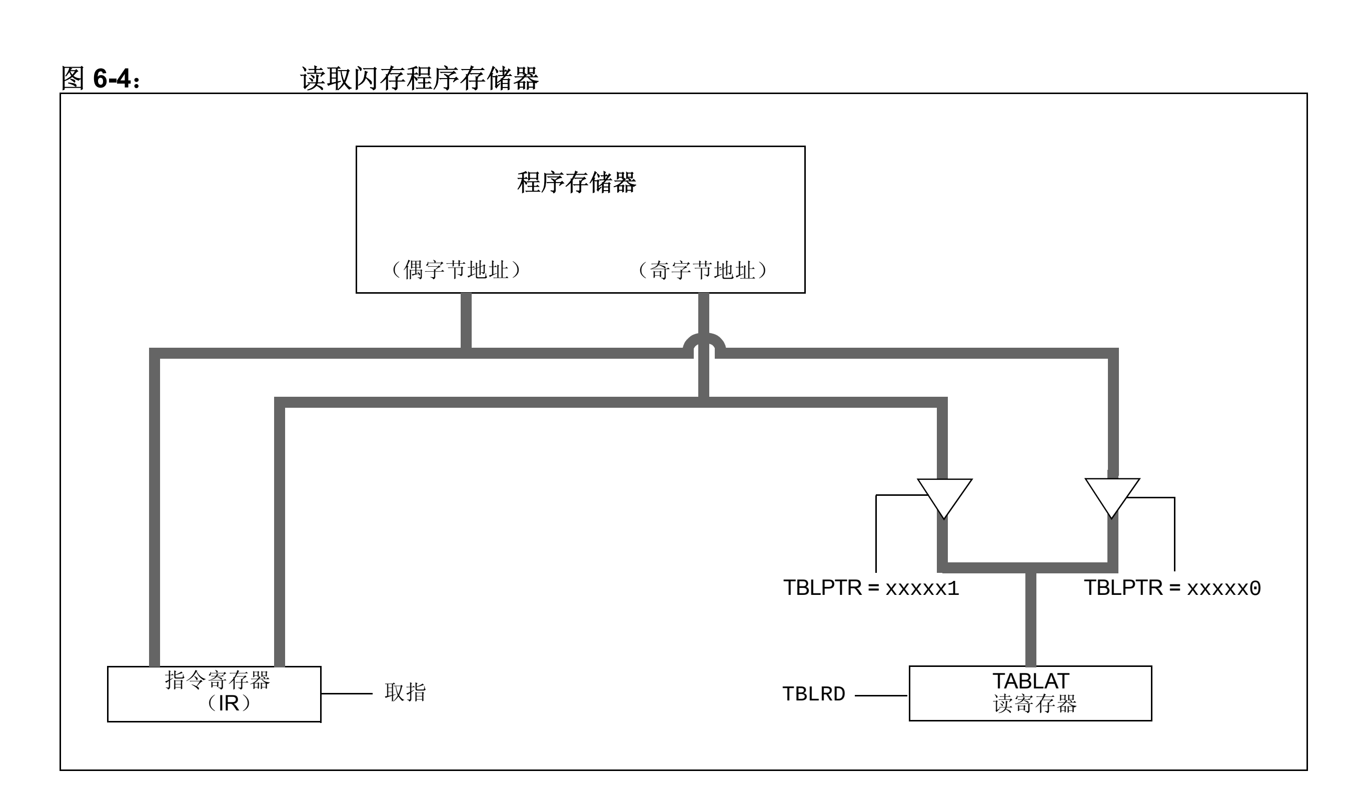
6.3 读取闪存程序存储器

TBLRD 指令用于从程序存储器获取数据并存入数据 RAM。表读操作每次从程序存储器读取一个字节。

TBLPTR 指向程序存储空间内的某个字节。执行 TBLRD 会将该字节存入TABLAT。另外，可以自动修改TBLPTR 以进行下次表读操作。

内部程序存储器通常是以字为单位构成的。由地址的最低有效位选择字的高字节或者低字节。图 6-4 显示了内部程序存储器和 TABLAT 之间的接口。

图 6-4： 读取闪存程序存储器



例 6-1： 读取一个闪存程序存储器字

```
        MOVLW   CODE_ADDR_UPPER         ; Load TBLPTR with the base
        MOVWF   TBLPTRU                 ; address of the word
        MOVLW   CODE_ADDR_HIGH
        MOVWF   TBLPTRH
        MOVLW   CODE_ADDR_LOW
        MOVWF   TBLPTRL
READ_WORD
        TBLRD*+                         ; read into TABLAT and increment
        MOVF    TABLAT, W               ; get data
        MOVWF   WORD_EVEN
        TBLRD*+                         ; read into TABLAT and increment
        MOVF    TABLAT, W               ; get data
        MOVF    WORD_ODD
```

6.4 擦除闪存程序存储器

最小擦除块大小为 32 字（即 64 字节）。只有通过使用外部编程器或通过 ICSP 控制，才能够批量擦除更大的程序存储器块。闪存阵列不支持字擦除。

当单片机本身启动擦除操作时，将擦除程序存储器的一个 64 字节块。TBLPTR<21:6> 的高 16 位指向将被擦除的块。 TBLPTR<5:0> 被忽略。

擦除操作由 EECON1 寄存器控制。必须将 EEPGD 位置 1 以指向闪存程序存储器。必须将 WREN 位置 1 以使能写操作。必须将 FREE 位置 1 以选择擦除操作。

为了保护数据，必须采用 EECON2 的写操作启动顺序。

要擦除内部闪存，必须有长写周期。在长写周期中，指令执行暂停。内部编程定时器将终止长周期写操作。

6.4.1 闪存程序存储器擦除顺序

擦除内部程序存储器块的事件顺序如下：

1. 将要擦除的行地址装入表指针寄存器；
2. 设置 EECON1 寄存器以执行擦除操作：
   - 将 EEPGD 位置 1 以指向程序存储器；
   - 将 CFGS 位清零以访问程序存储器；
   - 将 WREN 位置 1 以使能写操作；
   - 将 FREE 位置 1 以使能擦除操作。
3. 禁止中断。
4. 向 EECON2 写入 55h。
5. 向 EECON2 写入 0AAh。
6. 将 WR 位置 1。启动行擦除周期。
7. 在擦除操作期间， CPU 将停止工作 （内部定时器计时 2 ms 左右）。
8. 重新允许中断。

例 6-2： 擦除闪存程序存储器行

```
                MOVLW   CODE_ADDR_UPPER     ; load TBLPTR with the base
                MOVWF   TBLPTRU             ; address of the memory block
                MOVLW   CODE_ADDR_HIGH
                MOVWF   TBLPTRH
                MOVLW   CODE_ADDR_LOW
                MOVWF   TBLPTRL
        ERASE_ROW
                BSF     EECON1, EEPGD       ; point to Flash program memory
                BCF     EECON1, CFGS        ; access Flash program memory
                BSF     EECON1, WREN        ; enable write to memory
                BSF     EECON1, FREE        ; enable Row Erase operation
                BCF     INTCON, GIE         ; disable interrupts
    必需的序列   MOVLW   55h
                MOVWF   EECON2              ; write 55h
                MOVLW   0AAh
                MOVWF   EECON2              ; write 0AAh
                BSF     EECON1, WR          ; start erase (CPU stall)
                BSF     INTCON, GIE         ; re-enable interrupts
```


6.5 写入闪存程序存储器

最小编程块为 16 字（即 32 字节）。不支持字或字节编程。

表写操作用于将数据装入编程闪存存储器所需的保持寄存器。总共有 32 个保持寄存器用于表写编程。

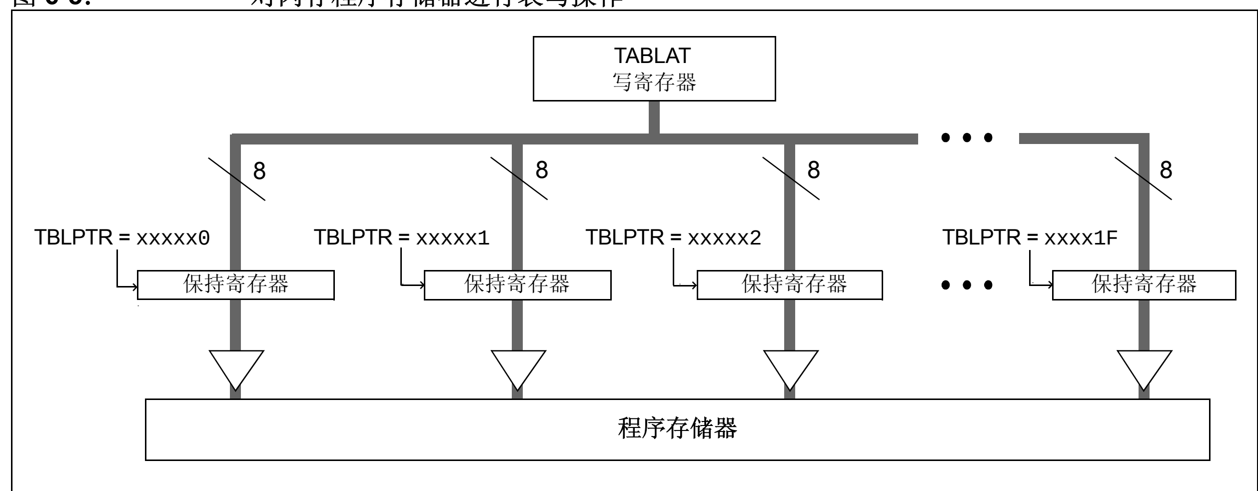
因为表锁存器 （TABLAT）是单字节，所以每次编程操作必须执行32次TBLWT指令。由于只写入保持寄存器，所有表写操作实际上都是短周期写操作。在更新完32个保持寄存器后，必须写入 EECON1 寄存器，以便启动长写周期编程操作。

要编程内部闪存，必须使用长写周期。在长写周期中，指令暂停执行。内部编程定时器将终止长写周期。

EEPROM 片上定时器控制写操作的时间。写 / 擦除电压由可在器件电压范围内运行的片上电荷泵产生。

> **注：** 器件复位时和写操作后保持寄存器的默认值为 FFh。向保持寄存器写入 FFh 并不会修改该寄存器的值。这意味着，仅修改程序存储器中的个别字节也是可以的。当修改个别字节时，在执行写操作前不必装载全部 32 个保持寄存器。

图 6-5： 对闪存程序存储器进行表写操作



6.5.1 闪存程序存储器写操作顺序

对内部程序存储器编程的事件顺序应该为：

1. 将 64 字节读入 RAM。
2. 必要时更新 RAM 中的数据。
3. 把要擦除的目标地址装入表指针寄存器。
4. 执行行擦除操作。
5. 把要写入的第一个字节的地址装入表指针寄存器。
6. 通过自动递增操作将32个字节写入保持寄存器。
7. 设置 EECON1 寄存器以执行写操作：
   - 将 EEPGD 位置 1 以指向程序存储器；
   - 将 CFGS 位清零以访问程序存储器；
   - 将 WREN 位置 1 以使能字节写入。
8. 禁止中断。
9. 向 EECON2 写入 55h。
10. 向 EECON2 写入 0AAh。
11. 将 WR 位置 1。启动写周期。
12. 在写操作期间 CPU 将停止工作 （内部定时器计时 2 ms 左右）。
13. 重新允许中断。
14. 重复步骤 6 到 14，共写入 64 个字节。
15. 验证存储器 （表读）。

更新存储器的一行 （64 字节）大约需要 8 ms。例 6-3 给出了所需的代码示例。

> **注：** 在 WR 位置 1 前，表指针需要指向 32 个保持寄存器。

例 6-3： 写闪存程序存储器

```
                MOVLW   D'64&#xD5;              ; number of bytes in erase block
                MOVWF   COUNTER
                MOVLW   BUFFER_ADDR_HIGH        ; point to buffer
                MOVWF   FSR0H
                MOVLW   BUFFER_ADDR_LOW
                MOVWF   FSR0L
                MOVLW   CODE_ADDR_UPPER         ; Load TBLPTR with the base
                MOVWF   TBLPTRU                 ; address of the memory block
                MOVLW   CODE_ADDR_HIGH
                MOVWF   TBLPTRH
                MOVLW   CODE_ADDR_LOW
                MOVWF   TBLPTRL
READ_BLOCK
                TBLRD*+                         ; read into TABLAT, and inc
                MOVF    TABLAT, W               ; get data
                MOVWF   POSTINC0                ; store data
                DECFSZ  COUNTER                 ; done?
                BRA     READ_BLOCK              ; repeat
MODIFY_WORD
                MOVLW   DATA_ADDR_HIGH          ; point to buffer
                MOVWF   FSR0H
                MOVLW   DATA_ADDR_LOW
                MOVWF   FSR0L
                MOVLW   NEW_DATA_LOW            ; update buffer word
                MOVWF   POSTINC0
                MOVLW   NEW_DATA_HIGH
                MOVWF   INDF0
ERASE_BLOCK
                MOVLW   CODE_ADDR_UPPER         ; load TBLPTR with the base
                MOVWF   TBLPTRU                 ; address of the memory block
                MOVLW   CODE_ADDR_HIGH
                MOVWF   TBLPTRH
                MOVLW   CODE_ADDR_LOW
                MOVWF   TBLPTRL
                BSF     EECON1, EEPGD           ; point to Flash program memory
                BCF     EECON1, CFGS            ; access Flash program memory
                BSF     EECON1, WREN            ; enable write to memory
                BSF     EECON1, FREE            ; enable Row Erase operation
                BCF     INTCON, GIE             ; disable interrupts
                MOVLW   55h
必需的          MOVWF   EECON2                  ; write 55h
序列            MOVLW   0AAh
                MOVWF   EECON2                  ; write 0AAh
                BSF     EECON1, WR              ; start erase (CPU stall)
                BSF     INTCON, GIE             ; re-enable interrupts
                TBLRD*-                         ; dummy read decrement
                MOVLW   BUFFER_ADDR_HIGH        ; point to buffer
                MOVWF   FSR0H
                MOVLW   BUFFER_ADDR_LOW
                MOVWF   FSR0L
                MOVLW   D&#xD5;2&#xD5;
                MOVWF   COUNTER1
WRITE_BUFFER_BACK
                MOVLW   D&#xD5;32&#xD5;            ; number of bytes in holding register
                MOVWF   COUNTER
WRITE_BYTE_TO_HREGS
                MOVF    POSTINC0, W             ; get low byte of buffer data
                MOVWF   TABLAT                  ; present data to table latch
                TBLWT+*                         ; write data, perform a short write
                                                ; to internal TBLWT holding register.
                DECFSZ  COUNTER                 ; loop until buffers are full
                BRA     WRITE_WORD_TO_HREGS
```

例 6-3： 写闪存程序存储器 （续）

```
PROGRAM_MEMORY
                BSF     EECON1, EEPGD          ; point to Flash program memory
                BCF     EECON1, CFGS           ; access Flash program memory
                BSF     EECON1, WREN           ; enable write to memory
                BCF     INTCON, GIE            ; disable interrupts
                MOVLW   55h
   必需的序列    MOVWF   EECON2                 ; write 55h
                MOVLW   0AAh
                MOVWF   EECON2                 ; write 0AAh
                BSF     EECON1, WR             ; start program (CPU stall)
                DECFSZ  COUNTER1
                BRA     WRITE_BUFFER_BACK
                BSF     INTCON, GIE            ; re-enable interrupts
                BCF     EECON1, WREN           ; disable write to memory
```

6.5.2 写校验

根据具体的应用，好的编程习惯一般要求使用原始值对写入值进行校验。当连续写入过多的数据已接近规范极限值时，就应该采用写校验。

6.5.3 写操作意外终止

如果由于意外事件 （如掉电和意外复位）终止了写操作，就应该对刚刚编程的存储单元进行验证，如有必要，还要重新进行编程。如果写入操作在正常操作过程中被 MCLR 复位或 WDT 超时复位中断，用户就要检测 WRERR 位并根据需要重新写入。

6.5.4 避免误写操作

为了防止误写闪存程序存储器，必须遵循写操作启动顺序。详情请参见**第 25.0 节 “CPU 的特殊性能”**。

6.6 代码保护时的闪存程序存储器操作

如需了解有关闪存程序存储器代码保护的具体信息，请参见**第 25.5 节 “程序校验和代码保护”**。

表 6-2： 与闪存程序存储器相关的寄存器

名称	Bit 7	Bit 6	Bit 5	Bit 4	Bit 3	Bit 2	Bit 1	Bit 0	复位值所在页
TBLPTRU	—	—	bit 21	程序存储器表指针最高字节 （TBLPTR<20:16>）					51
TBPLTRH	程序存储器表指针高字节 （TBLPTR<15:8>）								51
TBLPTRL	程序存储器表指针低字节 （TBLPTR<7:0>）								51
TABLAT	程序存储器表锁存器								51
INTCON	GIE/GIEH	PEIE/GIEL	TMR0IE	INT0IE	RBIE	TMR0IF	INT0IF	RBIF	51
EECON2	EEPROM 控制寄存器 2 （非物理寄存器）								53
EECON1	EEPGD	CFGS	—	FREE	WRERR	WREN	WR	RD	53
IPR2	OSCFIP	CMIP	USBIP	EEIP	BCLIP	HLVDIP	TMR3IP	CCP2IP	54
PIR2	OSCFIF	CMIF	USBIF	EEIF	BCLIF	HLVDIF	TMR3IF	CCP2IF	54
PIE2	OSCFIE	CMIE	USBIE	EEIE	BCLIE	HLVDIE	TMR3IE	CCP2IE	54

图注： — = 未用 （读为 0）。访问闪存 /EEPROM 存储器时不使用阴影单元。

注：

7.0 数据 EEPROM 存储器

不同于数据 RAM 和程序存储器，数据 EEPROM 是非易失性存储阵列，用来长期保存程序数据。EEPROM 并不直接映射到寄存器文件或程序存储器空间，而是通过特殊功能寄存器（SFR）来间接寻址。数据 EEPROM 在整个 VDD 范围内的正常运行期间是可读写的。

有 4 个 SFR 用于读写数据 EEPROM 和程序存储器。它们是：

- EECON1
- EECON2
- EEDATA
- EEADR

EEPROM 数据存储器允许按字节读写。在与数据存储器模块进行交互时，EEDATA 内存放 8 位读写数据，而 EEADR 寄存器存放要访问的 EEPROM 地址。

EEPROM 数据存储器具有很强的耐擦写能力。字节写操作将自动擦除该单元并写入新的值（即先擦后写）。写入时间由片上定时器控制。根据电压、温度和芯片的不同，写入时间也有差异。具体的规定请参见参数 D122（第 **28.0** 节 **"电气规范"**中的表 28-1）。

7.1 EECON1 和 EECON2 寄存器

EECON1 和 EECON2 寄存器控制对数据 EEPROM 的访问。这对寄存器也控制对程序存储器的访问，访问方式与访问数据 EEPROM 相同。

EECON1 寄存器（寄存器 7-1）用于控制对数据和程序存储器的访问。控制位 EEPGD 决定访问的是程序存储器还是数据 EEPROM。清零时，将访问数据 EEPROM 存储器。置 1 时，则访问程序存储器。

控制位 CFGS 决定访问的是配置寄存器还是程序存储器 / 数据 EEPROM。置 1 时，后续操作会针对配置寄存器。而当 CFGS 清零时，则由 EEPGD 位选择程序闪存存储器或数据 EEPROM 存储器。

将 WREN 位置 1 将允许一次写操作。上电时将清零 WREN 位。当 WREN 位置 1 时， WRERR 位将被硬件置 1 ；当在内部编程定时器超时且写操作完成时，WRERR 位将被清零。

注：	如果在正常工作期间， WRERR 读取值为 `1`，则表明写操作因复位而提早终止或进行了非法的写操作。

控制位 WR 用于启动写操作。此位只能由软件置 1 而不能清零。写操作完成后，由硬件将其清零。

注：	当写操作完成时EEIF中断标志位（PIR2<4>）置 1。此标志位必须由软件清零。

控制位 RD 和 WR 分别开始读取和擦 / 写操作。固件将这些位置 1，并在操作完成时由硬件清零。

在访问程序存储器（EEPGD = `1`）时，RD 位无法置 1。程序存储器是通过表读指令读取的。有关表读操作的信息，请参见第 **6.1** 节 **"表读和表写"**。

EECON2 寄存器不是物理寄存器。它专用于存储器的擦写操作。读 EECON2 得到的是全 `0`。

寄存器 7-1： EECON1：数据 EEPROM 控制寄存器 1

R/W-x	R/W-x	U-0	R/W-0	R/W-x	R/W-0	R/S-0	R/S-0
EEPGD	CFGS	—	FREE	WRERR	WREN	WR	RD
bit 7							bit 0

bit 7 **EEPGD**：闪存程序存储器或数据 EEPROM 选择位
1 = 访问闪存程序存储器
0 = 访问数据 EEPROM

bit 6 **CFGS**：闪存程序存储器 / 数据 EEPROM 或配置寄存器选择位
1 = 访问配置寄存器
0 = 访问闪存程序存储器或数据 EEPROM

bit 5 未用：读为 0

bit 4 **FREE**：闪存行擦除使能位
1 = 在下一条 WR 命令时擦除 TBLPTR 指向的程序存储器行 （擦除操作完成时清零）
0 = 仅执行写操作

bit 3 **WRERR**：闪存程序 / 数据 EEPROM 错误标志位
1 = 写操作提早终止 （由于正常工作中自定时编程期间的任何复位，或非法写入）
0 = 写操作完成

注： 当发生 WRERR 时， EEPGD 和 CFGS 位不会清零。
这样可以跟踪错误状态。

bit 2 **WREN**：闪存程序 / 数据 EEPROM 写使能位
1 = 允许写入闪存程序 / 数据 EEPROM
0 = 禁止写入闪存程序 / 数据 EEPROM

bit 1 **WR**：写控制位
1 = 启动数据 EEPROM 擦写周期或程序存储器擦写周期
（该操作是自定时的，一旦写入完成该位即由硬件清零。
WR 位只能由软件置 1 而不能由其清零。）
0 = 写入 EEPROM 的周期完成

bit 0 **RD**：读控制位
1 = 开始读 EEPROM
（读操作需要一个周期。硬件将 RD 位清零。 RD 位只能由软件置 1 而不能由其清零。当 EEPGD = 1 或 CFGS = 1 时 RD 不能置 1。）
0 = 未启动 EEPROM 读操作

图注：			
R = 可读位	W = 可写位	U = 未用位，读为 0	
-n = 上电复位时的值	1 = 置 1	0 = 清零	x = 未知

7.2 读数据 EEPROM 存储器

要读取数据存储器单元，用户必须将地址写入 EEADR 寄存器，清零 EEPGD 控制位（EECON1<7>）然后将控制位 RD 置 1（EECON1<0>）。由于该数据在下一个指令周期才可用；因此，EEDATA 寄存器可由下一条指令读取。 EEDATA 将保持这个值直到另一次读操作，或被用户写入（在写操作过程中）。

例 7-1 显示了读数据的基本过程。

7.3 写数据 EEPROM 存储器

要写 EEPROM 数据单元，必须首先将地址写入 EEADR 寄存器并将数据写入 EEDATA 寄存器。必须遵循例 7-2 中的执行顺序来启动写周期。

如果没有严格按照以上顺序（将 55h 写入 EECON2，将 0AAh 写入 EECON2，然后将 WR 位置 1）逐字节写入，写操作将不会开始。在该代码段执行过程中，强烈建议禁止中断。

此外，EECON1 中的 WREN 位也必须置 1 以使能写操作。这种机制可防止由于意外执行代码（即程序失控）造成对数据 EEPROM 的误写入。除了在更新 EEPROM 时， WREN 位将一直保持清零状态。 WREN 位不能由硬件清零。

写序列开始以后，就不能修改 EECON1、 EEADR 和 EDATA 了。除非将 WREN 位置 1，否则将不允许将 WR 位置 1。在前一指令中必须将 WREN 位置 1。 WR 和 WREN 位不能由同一指令置 1。

写周期完成后， WR 位将被硬件清零，同时 EEPROM 中断标志位（EEIF）被置 1。用户可以允许写中断或对 WR 位进行查询。 EEIF 必须由软件清零。

7.4 写校验

根据具体的应用，好的编程习惯一般要求使用原始值对写入值进行校验。当连续写入的数据字节数已接近规范极限时，就应该采用写校验。

例 7-1： 读数据 EEPROM

```
MOVLW   DATA_EE_ADDR      ;
MOVWF   EEADR             ; Lower bits of Data Memory Address to read
BCF     EECON1, EEPGD     ; Point to DATA memory
BCF     EECON1, CFGS      ; Access EEPROM
BSF     EECON1, RD        ; EEPROM Read
MOVF    EEDATA, W         ; W = EEDATA
```

例 7-2： 写数据 EEPROM

```
             MOVLW   DATA_EE_ADDR      ;
             MOVWF   EEADR             ; Lower bits of Data Memory Address to write
             MOVLW   DATA_EE_DATA      ;
             MOVWF   EEDATA            ; Data Memory Value to write
             BCF     EECON1, EPGD      ; Point to DATA memory
             BCF     EECON1, CFGS      ; Access EEPROM
             BSF     EECON1, WREN      ; Enable writes

             BCF     INTCON, GIE       ; Disable Interrupts
             MOVLW   55h               ;
必需的序列   MOVWF   EECON2            ; Write 55h
             MOVLW   0AAh              ;
             MOVWF   EECON2            ; Write 0AAh
             BSF     EECON1, WR        ; Set WR bit to begin write
             BSF     INTCON, GIE       ; Enable Interrupts

                                       ; User code execution
             BCF     EECON1, WREN      ; Disable writes on write complete (EEIF set)
```

7.5 代码保护时的操作

数据 EEPROM 在配置字中有它自己的代码保护位。如果使能了代码保护，将禁止对其进行外部读写操作。

单片机本身可以读写内部数据 EEPROM，与代码保护配置位的状态无关。更多详情请参见**第 25.0 节 “CPU 的特殊性能”**。

7.6 避免误写操作

有些情况下，用户可能不希望向数据 EEPROM 写入数据。为了防止误写 EEPROM，器件内建了各种保护机制。上电时，WREN 位被清零。此外，上电延时期间（TPWRT，表 28-12 中的参数 33）也禁止写 EEPROM。

在欠压、电源故障或软件故障期间，写操作启动序列和 WREN 位可共同防止误写操作的发生。

7.7 使用数据 EEPROM

数据 EEPROM 是高耐擦写、按字节寻址的阵列，特别适用于存储频繁更改的信息（如程序变量或其他经常更新的数据）。频繁更改值的更新频率通常要高于 D124 或 D124A 规范中的规定。如果情况并非如此，就必须执行阵列刷新。因此，不常修改的变量（例如常数、ID 和校准值等）应该存储在闪存程序存储器上。

例 7-3 所示为简单的数据 EEPROM 刷新程序。

注：	如果数据 EEPROM 仅用于存储常数和 / 或很少修改的数据，可能不必进行阵列刷新。请参见规范 D124 或 D124A。

例 7-3：　数据 EEPROM 刷新程序

```
              CLRF    EEADR              ; Start at address 0
              BCF     EECON1, CFGS       ; Set for memory
              BCF     EECON1, EEPGD      ; Set for Data EEPROM
              BCF     INTCON, GIE        ; Disable interrupts
              BSF     EECON1, WREN       ; Enable writes
      Loop                               ; Loop to refresh array
              BSF     EECON1, RD         ; Read current address
              MOVLW   55h                ;
  必需的序列  MOVWF   EECON2             ; Write 55h
              MOVLW   0AAh               ;
              MOVWF   EECON2             ; Write 0AAh
              BSF     EECON1, WR         ; Set WR bit to begin write
              BTFSC   EECON1, WR         ; Wait for write to complete
              BRA     $-2
              INCFSZ  EEADR, F           ; Increment address
              BRA     LOOP               ; Not zero, do it again

              BCF     EECON1, WREN       ; Disable writes
              BSF     INTCON, GIE        ; Enable interrupts
```

表 7-1： 与数据 EEPROM 存储器相关的寄存器

名称	Bit 7	Bit 6	Bit 5	Bit 4	Bit 3	Bit 2	Bit 1	Bit 0	复位值所在页
INTCON	GIE/GIEH	PEIE/GIEL	TMR0IE	INT0IE	RBIE	TMR0IF	INT0IF	RBIF	51
EEADR	EEPROM 地址寄存器								53
EEDATA	EEPROM 数据寄存器								53
EECON2	EEPROM 控制寄存器 2 （非物理寄存器）								53
EECON1	EEPGD	CFGS	—	FREE	WRERR	WREN	WR	RD	53
IPR2	OSCFIP	CMIP	USBIP	EEIP	BCLIP	HLVDIP	TMR3IP	CCP2IP	54
PIR2	OSCFIF	CMIF	USBIF	EEIF	BCLIF	HLVDIF	TMR3IF	CCP2IF	54
PIE2	OSCFIE	CMIE	USBIE	EEIE	BCLIE	HLVDIE	TMR3IE	CCP2IE	54

图注： — = 未用，读为 0。访问闪存 /EEPROM 存储器时不使用阴影单元。

注：

8.0 8 x 8 硬件乘法器

8.1 简介

所有的 PIC18 器件均包含一个 8 x 8 硬件乘法器（乘法器是 ALU 的一部分）。该乘法器可执行无符号运算并产生一个 16 位运算结果，该结果存储在一对乘积寄存器 PRODH:PRODL 中。该乘法器执行的运算不会影响 Status 寄存器中的任何标志。

通过硬件执行乘法运算只需要 1 个指令周期。硬件乘法器具有更高的计算吞吐量并减少了乘法算法的代码长度，从而可在许多先前仅能使用数字信号处理器的应用中使用 PIC18 器件。表 8-1 给出了各种硬件和软件乘法运算的比较，包括所需的存储空间和执行时间。

8.2 工作原理

例 8-1 给出了一个 8 x 8 无符号乘法运算的指令序列。当已在 WREG 寄存器中装入了一个乘数时，实现该运算仅需一条指令。

例 8-2 给出了执行 8 x 8 有符号乘法运算的指令序列。要弄清乘数的符号位，必须检查每个乘数的最高有效位（MSb），并做相应的减法。

例 **8-1**： **8 x 8** 无符号乘法程序

```
MOVF    ARG1, W      ;
MULWF   ARG2         ; ARG1 * ARG2 ->
                     ; PRODH:PRODL
```

例 **8-2**： **8 x 8** 有符号乘法程序

```
MOVF    ARG1, W
MULWF   ARG2         ; ARG1 * ARG2 ->
                     ; PRODH:PRODL
BTFSC   ARG2, SB     ; Test Sign Bit
SUBWF   PRODH, F     ; PRODH = PRODH
                     ;         - ARG1
MOVF    ARG2, W
BTFSC   ARG1, SB     ; Test Sign Bit
SUBWF   PRODH, F     ; PRODH = PRODH
                     ;         - ARG2
```

表 **8-1**： 各种乘法运算的性能比较

程序	乘法实现方法	程序存储空间（字数）	周期数（最多）	时间		
				@40 MHz	**@10 MHz**	**@4 MHz**
8 x 8 无符号	软件乘法	13	69	6.9 μs	27.6 μs	69 μs
	硬件乘法	1	1	100 ns	400 ns	1 μs
8 x 8 有符号	软件乘法	33	91	9.1 μs	36.4 μs	91 μs
	硬件乘法	6	6	600 ns	2.4 μs	6 μs
16 x 16 无符号	软件乘法	21	242	24.2 μs	96.8 μs	242 μs
	硬件乘法	28	28	2.8 μs	11.2 μs	28 μs
16 x 16 有符号	软件乘法	52	254	25.4 μs	102.6 μs	254 μs
	硬件乘法	35	40	4.0 μs	16.0 μs	40 μs

例 8-3 给出了一个 16 × 16 无符号乘法运算的指令序列。公式 8-1 为所使用的算法。32 位运算结果存储在 4 个寄存器（RES3:RES0）中。

公式 8-1： 16 x 16 无符号乘法算法

$$
\begin{aligned}
RES3:RES0 &= ARG1H:ARG1L \bullet ARG2H:ARG2L \\
&= (ARG1H \bullet ARG2H \bullet 2^{16}) + \\
&\quad (ARG1H \bullet ARG2L \bullet 2^{8}) + \\
&\quad (ARG1L \bullet ARG2H \bullet 2^{8}) + \\
&\quad (ARG1L \bullet ARG2L)
\end{aligned}
$$

例 8-3： 16 x 16 无符号乘法程序

```
    MOVF    ARG1L, W
    MULWF   ARG2L           ; ARG1L * ARG2L->
                            ; PRODH:PRODL
    MOVFF   PRODH, RES1     ;
    MOVFF   PRODL, RES0     ;
;
    MOVF    ARG1H, W
    MULWF   ARG2H           ; ARG1H * ARG2H->
                            ; PRODH:PRODL
    MOVFF   PRODH, RES3     ;
    MOVFF   PRODL, RES2     ;
;
    MOVF    ARG1L, W
    MULWF   ARG2H           ; ARG1L * ARG2H->
                            ; PRODH:PRODL
    MOVF    PRODL, W        ;
    ADDWF   RES1, F         ; Add cross
    MOVF    PRODH, W        ; products
    ADDWFC  RES2, F         ;
    CLRF    WREG            ;
    ADDWFC  RES3, F         ;
;
    MOVF    ARG1H, W        ;
    MULWF   ARG2L           ; ARG1H * ARG2L->
                            ; PRODH:PRODL
    MOVF    PRODL, W        ;
    ADDWF   RES1, F         ; Add cross
    MOVF    PRODH, W        ; products
    ADDWFC  RES2, F         ;
    CLRF    WREG            ;
    ADDWFC  RES3, F         ;
```

例 8-4 给出了 16 × 16 有符号乘法运算的指令序列。公式 8-2 为所使用的算法。32 位运算结果存储在 4 个寄存器（RES3:RES0）中。要弄清乘数的符号位，必须检查每个乘数对的最高有效位（MSb），并做相应的减法。

公式 8-2： 16 x 16 有符号乘法算法

$$
\begin{aligned}
RES3:RES0 &= ARG1H:ARG1L \bullet ARG2H:ARG2L \\
&= (ARG1H \bullet ARG2H \bullet 2^{16}) + \\
&\quad (ARG1H \bullet ARG2L \bullet 2^{8}) + \\
&\quad (ARG1L \bullet ARG2H \bullet 2^{8}) + \\
&\quad (ARG1L \bullet ARG2L) + \\
&\quad (-1 \bullet ARG2H<7> \bullet ARG1H:ARG1L \bullet 2^{16}) + \\
&\quad (-1 \bullet ARG1H<7> \bullet ARG2H:ARG2L \bullet 2^{16})
\end{aligned}
$$

例 8-4： 16 x 16 有符号乘法程序

```
    MOVF    ARG1L, W
    MULWF   ARG2L           ; ARG1L * ARG2L ->
                            ; PRODH:PRODL
    MOVFF   PRODH, RES1     ;
    MOVFF   PRODL, RES0     ;
;
    MOVF    ARG1H, W
    MULWF   ARG2H           ; ARG1H * ARG2H ->
                            ; PRODH:PRODL
    MOVFF   PRODH, RES3     ;
    MOVFF   PRODL, RES2     ;
;
    MOVF    ARG1L,W
    MULWF   ARG2H           ; ARG1L * ARG2H ->
                            ; PRODH:PRODL
    MOVF    PRODL, W        ;
    ADDWF   RES1, F         ; Add cross
    MOVF    PRODH, W        ; products
    ADDWFC  RES2, F         ;
    CLRF    WREG            ;
    ADDWFC  RES3, F         ;
;
    MOVF    ARG1H, W        ;
    MULWF   ARG2L           ; ARG1H * ARG2L ->
                            ; PRODH:PRODL
    MOVF    PRODL, W        ;
    ADDWF   RES1, F         ; Add cross
    MOVF    PRODH, W        ; products
    ADDWFC  RES2, F         ;
    CLRF    WREG            ;
    ADDWFC  RES3, F         ;
;
    BTFSS   ARG2H, 7        ; ARG2H:ARG2L neg?
    BRA     SIGN_ARG1       ; no, check ARG1
    MOVF    ARG1L, W        ;
    SUBWF   RES2            ;
    MOVF    ARG1H, W        ;
    SUBWFB  RES3
;
SIGN_ARG1
    BTFSS   ARG1H, 7        ; ARG1H:ARG1L neg?
    BRA     CONT_CODE       ; no, done
    MOVF    ARG2L, W        ;
    SUBWF   RES2            ;
    MOVF    ARG2H, W        ;
    SUBWFB  RES3
;
CONT_CODE
   :
```


初稿

9.0 中断

PIC18F2455/2550/4455/4550 器件提供多个中断源及一个中断优先级功能，可以给大多数中断源分配高优先级或者低优先级。高优先级中断向量地址为 000008h，低优先级中断向量地址为 000018h。高优先级中断事件将中断所有可能正在进行的低优先级中断。

有 10 个寄存器用于控制中断的操作。它们是：

- RCON
- INTCON
- INTCON2
- INTCON3
- PIR1 和 PIR2
- PIE1 和 PIE2
- IPR1 和 IPR2

建议使用由 MPLAB® IDE 提供的 Microchip 头文件命名这些寄存器中的位。这使得汇编器 / 编译器能够自动识别指定寄存器内这些位的位置。

通常，用三个位来控制中断源的操作。它们是：

- 标志位表明发生了中断事件
- 允许位允许程序跳转到中断向量地址处执行（当标志位置 1 时）
- 优先级位用于选择是高优先级还是低优先级

通过将 IPEN 位（RCON<7>）置 1，可使能中断优先级功能。当使能了中断优先级时，有 2 位可允许全局中断。将 GIEH 位（INTCON<7>）置 1，可允许所有优先级位置 1（高优先级）的中断。将 GIEL 位（INTCON<6>）置 1，可允许所有优先级位清零（低优先级）的中断。当中断标志位、允许位以及相应的全局中断允许位均被置 1 时，程序将立即跳转到中断地址 000008h 或 000018h，具体地址取决于优先级位的设置。通过设置相应的允许位可以禁止单个中断。

当 IPEN 位被清零（默认状态）时，便会禁止中断优先级功能，此时中断与 PICmicro® 中档器件的相兼容。在兼容模式下，各个中断源的中断优先级位均不起作用。INTCON<6> 是 PEIE 位，它可允许 / 禁止所有的外设中断源。 INTCON<7> 是 GIE 位，它可允许 / 禁止所有的中断源。在兼容模式下，所有中断均跳转到地址 000008h。

当响应中断时，全局中断允许位被清零以禁止其他中断。如果清零 IPEN 位，全局中断允许位就是 GIE 位。如果使用中断优先级，这个位将是 GIEH 位或者 GIEL 位。高优先级中断源会中断低优先级中断。处理高优先级中断时，低优先级中断将不被响应。

返回地址被压入堆栈，PC 中装入中断向量地址（000008h 或 000018h）。进入中断服务程序之后，就可以通过查询中断标志位来确定中断源。在重新允许中断前，必须用软件将中断标志位清零，以避免重复响应该中断。

执行 “中断返回” 指令 `RETFIE`，退出中断程序并将 GIE 位（若使用中断优先级，则为 GIEH 或 GIEL 位）置 1，以重新允许中断。

对于外部中断事件，诸如 INT 引脚中断或者 PORTB 输入电平变化中断，中断响应延时将会是 3 到 4 个指令周期。对于单周期或双周期指令，中断响应延时完全相同。不管对应的中断允许位和 GIE 位状态如何，各中断标志位均被置 1。

注：	当**任何**中断被允许时，不要使用 `MOVFF` 指令来修改中断控制寄存器。否则可能引起单片机操作出错。

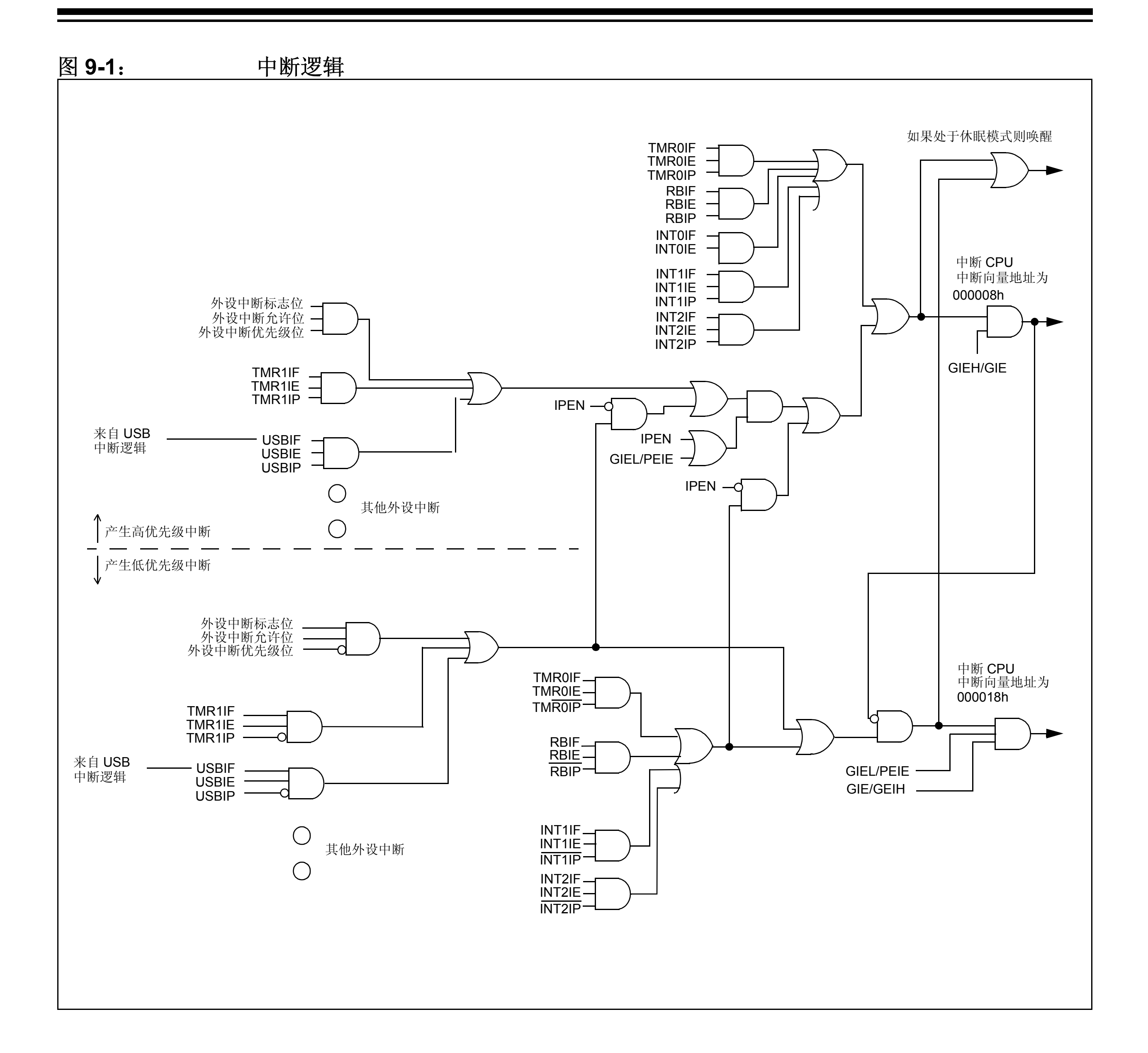
9.1 USB 中断

与其他外设不同，USB 模块可由多类事件产生中断。这些事件包括正常通信事件、状态事件和模块电平错误事件。

USB 模块配备了自己的中断逻辑以处理这些事件。其中断逻辑的处理方式类似于单片机电平中断，每个中断源都有独立的标志位和允许位。所有事件是否发生均反映在器件电平中断标志位 USBIF（PIR2<5>）上。不像器件电平中断逻辑，每个 USB 中断事件不能单独的分配各自的优先级。它可在设置器件电平中断优先级时由 USB 事件中断优先级位 USBIP 统一设定。

关于 USB 中断逻辑的更多信息，请参见**第 17.5 节 “USB 中断”**。

图 9-1： 中断逻辑



TMR0IF
TMR0IE
TMR0IP
RBIF
RBIE
RBIP
INT0IF
INT0IE
INT1IF
INT1IE
INT1IP
INT2IF
INT2IE
INT2IP
如果处于休眠模式则唤醒
中断 CPU
中断向量地址为
000008h
GIEH/GIE
外设中断标志位
外设中断允许位
外设中断优先级位
TMR1IF
TMR1IE
TMR1IP
来自 USB
中断逻辑
USBIF
USBIE
USBIP
IPEN
GIEL/PEIE
其他外设中断
产生高优先级中断
产生低优先级中断
中断 CPU
中断向量地址为
000018h
GIEL/PEIE
GIE/GEIH

 初稿

9.2 INTCON 寄存器

INTCON 寄存器是可读写的寄存器，包含多个允许位、优先级位和标志位。

> **注：** 当中断条件产生时，不管相应的中断允许位或全局中断允许位的状态如何，中断标志位都将置 1。用户软件应在允许一个中断之前，确保先将该中断标志位清零。故中断标志位可以用于软件查询。

寄存器 9-1： INTCON：中断控制寄存器

R/W-0	R/W-0	R/W-0	R/W-0	R/W-0	R/W-0	R/W-0	R/W-x
GIE/GIEH	PEIE/GIEL	TMR0IE	INT0IE	RBIE	TMR0IF	INT0IF	RBIF
bit 7							bit 0

bit 7 **GIE/GIEH**：全局中断允许位
当 IPEN = 0 时：
1 = 允许所有未屏蔽的中断
0 = 禁止所有中断
当 IPEN = 1 时：
1 = 允许所有高优先级中断
0 = 禁止所有高优先级中断

bit 6 **PEIE/GIEL**：外设中断允许位
当 IPEN = 0 时：
1 = 允许所有未屏蔽的外设中断
0 = 禁止所有外设中断
当 IPEN = 1 时：
1 = 允许所有低优先级的外设中断
0 = 禁止所有低优先级的外设中断

bit 5 **TMR0IE**：TMR0 溢出中断允许位
1 = 允许 TMR0 溢出中断
0 = 禁止 TMR0 溢出中断

bit 4 **INT0IE**：INT0 外部中断允许位
1 = 允许 INT0 外部中断
0 = 禁止 INT0 外部中断

bit 3 **RBIE**：RB 端口电平变化中断允许位
1 = 允许 RB 端口电平变化中断
0 = 禁止 RB 端口电平变化中断

bit 2 **TMR0IF**：TMR0 溢出中断标志位
1 = TMR0 寄存器已发生溢出 （必须由软件清零）
0 = TMR0 寄存器未发生溢出

bit 1 **INT0IF**：INT0 外部中断标志位
1 = 发生了 INT0 外部中断 （必须由软件清零）
0 = 未发生 INT0 外部中断

bit 0 **RBIF**：RB 端口电平变化中断标志位
1 = RB7:RB4 引脚中至少有一个引脚的电平状态发生了改变 （必须由软件清零）
0 = RB7:RB4 引脚的电平状态没有改变

> **注：** 引脚上电平变化会一直不断地将此位置 1。读取 PORTB 可以结束这种情况，并将该位清零。

图注：			
R = 可读位	W = 可写位	U= 未用位，读为 0	
-n = 上电复位时的值	1 = 置 1	0 = 清零	x = 未知

寄存器 9-2： INTCON2：中断控制寄存器 2

R/W-1	R/W-1	R/W-1	R/W-1	U-0	R/W-1	U-0	R/W-1
$\overline{\text{RBPU}}$	INTEDG0	INTEDG1	INTEDG2	—	TMR0IP	—	RBIP
bit 7							bit 0

bit 7 **$\overline{\text{RBPU}}$**：PORTB 上拉使能位
1 = 禁止所有 PORTB 上拉
0 = 按各个端口锁存值使能 PORTB 上拉

bit 6 **INTEDG0**：外部中断 0 边沿选择位
1 = 上升沿触发中断
0 = 下降沿触发中断

bit 5 **INTEDG1**：外部中断 1 边沿选择位
1 = 上升沿触发中断
0 = 下降沿触发中断

bit 4 **INTEDG2**：外部中断 2 边沿选择位
1 = 上升沿触发中断
0 = 下降沿触发中断

bit 3 未用：读为 0

bit 2 **TMR0IP**：TMR0 溢出中断优先级位
1 = 高优先级
0 = 低优先级

bit 1 未用：读为 0

bit 0 **RBIP**：RB 端口电平变化中断优先级位
1 = 高优先级
0 = 低优先级

图注：			
R = 可读位	W = 可写位	U = 未用位，读为 0	
-n = 上电复位时的值	1 = 置 1	0 = 清零	x = 未知

注： 当中断条件产生时，不管相应的中断允许位或全局中断允许位的状态如何，中断标志位都将置 1。用户软件应在允许一个中断之前，确保先将该中断标志位清零。故中断标志位可以用于软件查询。

寄存器 9-3： INTCON3：中断控制寄存器 3

R/W-1	R/W-1	U-0	R/W-0	R/W-0	U-0	R/W-0	R/W-0
INT2IP	INT1IP	—	INT2IE	INT1IE	—	INT2IF	INT1IF
bit 7							bit 0

bit 7 **INT2IP**：INT2 外部中断优先级位
1 = 高优先级
0 = 低优先级

bit 6 **INT1IP**：INT1 外部中断优先级位
1 = 高优先级
0 = 低优先级

bit 5 未用：读为 0

bit 4 **INT2IE**：INT2 外部中断允许位
1 = 允许 INT2 外部中断
0 = 禁止 INT2 外部中断

bit 3 **INT1IE**：INT1 外部中断允许位
1 = 允许 INT1 外部中断
0 = 禁止 INT1 外部中断

bit 2 未用：读为 0

bit 1 **INT2IF**：INT2 外部中断标志位
1 = 发生了 INT2 外部中断 （必须由软件清零）
0 = 未发生 INT2 外部中断

bit 0 **INT1IF**：INT1 外部中断标志位
1 = 发生了 INT1 外部中断 （必须由软件清零）
0 = 未发生 INT1 外部中断

图注：			
R = 可读位	W = 可写位	U = 未用位，读为 0	
-n = 上电复位时的值	1 = 置 1	0 = 清零	x = 未知

注：	当中断条件产生时，不管相应的中断允许位或全局中断允许位的状态如何，中断标志位都将置 1。用户软件应在允许一个中断之前，确保先将该中断标志位清零。故中断标志位可以用于软件查询。

9.3 PIR 寄存器

PIR 寄存器包含各外设中断的标志位。根据外设中断源的数量，有两个外设中断请求 （标志）寄存器 （PIR1 和 PIR2）。

> 注 1：当有中断条件产生时，不管对应的中断允许位或全局允许位 GIE（INTCON<7>）的状态如何，中断标志位都将置 1。
>
> 2：用户软件应在允许一个中断之前，确保先将该中断标志位清零；同时在响应该中断后，也应该将该中断标志位清零。

寄存器 9-4： PIR1：外设中断请求 （标志）寄存器 1

R/W-0	R/W-0	R-0	R-0	R/W-0	R/W-0	R/W-0	R/W-0
SPPIF[(1)]	ADIF	RCIF	TXIF	SSPIF	CCP1IF	TMR2IF	TMR1IF
bit 7							bit 0

bit 7 **SPPIF**：并行通信端口读 / 写中断标志位 [(1)]
1 = 发生了读 / 写操作 （必须由软件清零）
0 = 未发生读 / 写操作

注 1：此位在 28 引脚器件上是保留的，总是读为 0。

bit 6 **ADIF**：A/D 转换器中断标志位
1 = 一次 A/D 转换已完成 （必须由软件清零）
0 = A/D 转换未完成

bit 5 **RCIF**：EUSART 接收中断标志位
1 = EUSART 接收缓冲器 RCREG 已满 （当读取 RCREG 时清零）
0 = EUSART 接收缓冲器为空

bit 4 **TXIF**：EUSART 发送中断标志位
1 = EUSART 发送缓冲器 TXREG 为空 （当写入 TXREG 时清零）
0 = EUSART 发送缓冲器已满

bit 3 **SSPIF**：主控同步串行端口中断标志位
1 = 发送 / 接收已完成 （必须由软件清零）
0 = 等待发送 / 接收

bit 2 **CCP1IF**：CCP1 中断标志位
捕捉模式：
1 = 发生了 TMR1 寄存器捕捉（必须由软件清零）
0 = 未发生 TMR1 寄存器捕捉
比较模式：
1 = 发生了 TMR1 寄存器的比较匹配 （必须由软件清零）
0 = 未发生 TMR1 寄存器的比较匹配
PWM 模式：
在此模式下未使用。

bit 1 **TMR2IF**：TMR2 与 PR2 匹配中断标志位
1 = TMR2 与 PR2 发生匹配（必须由软件清零）
0 = TMR2 与 PR2 未发生匹配

bit 0 **TMR1IF**：TMR1 溢出中断标志位
1 = TMR1 寄存器已发生溢出 （必须由软件清零）
0 = TMR1 寄存器未发生溢出

图注：			
R = 可读位	W = 可写位	U = 未用位，读为 0	
-n = 上电复位时的值	1 = 置 1	0 = 清零	x = 未知

寄存器 9-5： PIR2：外设中断请求 （标志）寄存器 2

R/W-0	R/W-0	R/W-0	R/W-0	R/W-0	R/W-0	R/W-0	R/W-0
OSCFIF	CMIF	USBIF	EEIF	BCLIF	HLVDIF	TMR3IF	CCP2IF
bit 7							bit 0

bit 7 **OSCFIF**：振荡器故障中断标志位
1 = 器件振荡器发生故障，改由 INTOSC 提供时钟 （必须由软件清零）
0 = 系统时钟正常运行

bit 6 **CMIF**：比较器中断标志位
1 = 比较器输入已改变 （必须由软件清零）
0 = 比较器输入未发生变化

bit 5 **USBIF**：USB 中断标志位
1 = USB 请求中断 （必须由软件清零）
0 = 没有 USB 中断请求

bit 4 **EEIF**：数据 EEPROM/ 闪存写操作中断标志位
1 = 写操作完成 （必须由软件清零）
0 = 写操作未完成或还未开始

bit 3 **BCLIF**：总线冲突中断标志位
1 = 发生了总线冲突 （必须由软件清零）
0 = 未发生总线冲突

bit 2 **HLVDIF**：高 / 低压检测中断标志位
1 = 出现了高 / 低压条件 （必须由软件清零）
0 = 未出现高 / 低压条件

bit 1 **TMR3IF**：TMR3 溢出中断标志位
1 = TMR3 寄存器已溢出 （必须由软件清零）
0 = TMR3 寄存器未溢出

bit 0 **CCP2IF**：CCP2 中断标志位
捕捉模式：
1 = 发生了 TMR1 或 TMR3 寄存器捕捉 （必须由软件清零）
0 = 未发生 TMR1 或 TMR3 寄存器捕捉
比较模式：
1 = TMR1 或 TMR3 寄存器发生了比较匹配 （必须由软件清零）
0 = TMR1 或 TMR3 寄存器未发生比较匹配
PWM 模式：
在此模式下未使用。

图注：			
R = 可读位	W = 可写位	U = 未用位，读为 0	
-n = 上电复位时的值	1 = 置 1	0 = 清零	x = 未知

9.4 PIE 寄存器

PIE 寄存器包含各外设中断的允许位。根据外设中断源的数量，有两个外设中断允许寄存器（PIE1 和 PIE2）。当 IPEN = 0 时，要允许任何外设中断就必须将 PEIE 位置 1。

寄存器 9-6： PIE1：外设中断允许寄存器 1

R/W-0	R/W-0	R/W-0	R/W-0	R/W-0	R/W-0	R/W-0	R/W-0
SPPIE(1)	ADIE	RCIE	TXIE	SSPIE	CCP1IE	TMR2IE	TMR1IE
bit 7							bit 0

bit 7 **SPPIE**：并行通信端口读 / 写中断允许位 (1)
1 = 允许 SPP 读 / 写中断
0 = 禁止 SPP 读 / 写中断

注 1：此位在 28 引脚器件上是保留的，总是读为 0。

bit 6 **ADIE**：A/D 转换器中断允许位
1 = 允许 A/D 中断
0 = 禁止 A/D 中断

bit 5 **RCIE**：EUSART 接收中断允许位
1 = 允许 EUSART 接收中断
0 = 禁止 EUSART 接收中断

bit 4 **TXIE**：EUSART 发送中断允许位
1 = 允许 EUSART 发送中断
0 = 禁止 EUSART 发送中断

bit 3 **SSPIE**：主控同步串行端口中断允许位
1 = 允许 MSSP 中断
0 = 禁止 MSSP 中断

bit 2 **CCP1IE**：CCP1 中断允许位
1 = 允许 CCP1 中断
0 = 禁止 CCP1 中断

bit 1 **TMR2IE**：TMR2 与 PR2 匹配中断允许位
1 = 允许 TMR2 与 PR2 匹配中断
0 = 禁止 TMR2 与 PR2 匹配中断

bit 0 **TMR1IE**：TMR1 溢出中断允许位
1 = 允许 TMR1 溢出中断
0 = 禁止 TMR1 溢出中断

图注：			
R = 可读位	W = 可写位	U = 未用位，读为 0	
-n = 上电复位时的值	1 = 置 1	0 = 清零	x = 未知

寄存器 9-7： PIE2：外设中断允许寄存器 2

R/W-0	R/W-0	R/W-0	R/W-0	R/W-0	R/W-0	R/W-0	R/W-0
OSCFIE	CMIE	USBIE	EEIE	BCLIE	HLVDIE	TMR3IE	CCP2IE
bit 7							bit 0

bit 7 **OSCFIE**：振荡器故障中断允许位
1 = 允许
0 = 禁止

bit 6 **CMIE**：比较器中断允许位
1 = 允许
0 = 禁止

bit 5 **USBIE**：USB 中断允许位
1 = 允许
0 = 禁止

bit 4 **EEIE**：数据 EEPROM/ 闪存写操作中断允许位
1 = 允许
0 = 禁止

bit 3 **BCLIE**：总线冲突中断允许位
1 = 允许
0 = 禁止

bit 2 **HLVDIE**：高 / 低压检测中断允许位
1 = 允许
0 = 禁止

bit 1 **TMR3IE**：TMR3 溢出中断允许位
1 = 允许
0 = 禁止

bit 0 **CCP2IE**：CCP2 中断允许位
1 = 允许
0 = 禁止

图注：			
R = 可读位	W = 可写位	U = 未用位，读为 0	
-n = 上电复位时的值	1 = 置 1	0 = 清零	x = 未知

9.5 IPR 寄存器

IPR 寄存器包含各外设中断的优先级位。根据外设中断源的数量，有两个外设中断优先级寄存器（IPR1 和 IPR2）。使用优先级位要求将中断优先级使能（IPEN）位置 1。

寄存器 9-8： IPR1：外设中断优先级寄存器 1

R/W-1	R/W-1	R/W-1	R/W-1	R/W-1	R/W-1	R/W-1	R/W-1
SPPIP[(1)]	ADIP	RCIP	TXIP	SSPIP	CCP1IP	TMR2IP	TMR1IP
bit 7							bit 0

bit 7 **SPPIP** 并行通信端口读 / 写中断优先级位 [(1)]
1 = 高优先级
0 = 低优先级

注 1：此位在 28 引脚器件上是保留的，总是读为 0。

bit 6 **ADIP**：A/D 转换器中断优先级位
1 = 高优先级
0 = 低优先级

bit 5 **RCIP**：EUSART 接收中断优先级位
1 = 高优先级
0 = 低优先级

bit 4 **TXIP**：EUSART 发送中断优先级位
1 = 高优先级
0 = 低优先级

bit 3 **SSPIP**：主控同步串行端口中断优先级位
1 = 高优先级
0 = 低优先级

bit 2 **CCP1IP**：CCP1 中断优先级位
1 = 高优先级
0 = 低优先级

bit 1 **TMR2IP**：TMR2 与 PR2 匹配中断优先级位
1 = 高优先级
0 = 低优先级

bit 0 **TMR1IP**：TMR1 溢出中断优先级位
1 = 高优先级
0 = 低优先级

图注：			
R = 可读位	W = 可写位	U = 未用位，读为 0	
-n = 上电复位时的值	1 = 置 1	0 = 清零	x = 未知

寄存器 9-9： IPR2：外设中断优先级寄存器 2

R/W-1	R/W-1	R/W-1	R/W-1	R/W-1	R/W-1	R/W-1	R/W-1
OSCFIP	CMIP	USBIP	EEIP	BCLIP	HLVDIP	TMR3IP	CCP2IP
bit 7							bit 0

bit 7 **OSCFIP**：振荡器故障中断优先级位
1 = 高优先级
0 = 低优先级

bit 6 **CMIP**：比较器中断优先级位
1 = 高优先级
0 = 低优先级

bit 5 **USBIP**：USB 中断优先级位
1 = 高优先级
0 = 低优先级

bit 4 **EEIP**：数据 EEPROM/ 闪存写操作中断优先级位
1 = 高优先级
0 = 低优先级

bit 3 **BCLIP**：总线冲突中断优先级位
1 = 高优先级
0 = 低优先级

bit 2 **HLVDIP**：高 / 低压检测中断优先级位
1 = 高优先级
0 = 低优先级

bit 1 **TMR3IP**：TMR3 溢出中断优先级位
1 = 高优先级
0 = 低优先级

bit 0 **CCP2IP**：CCP2 中断优先级位
1 = 高优先级
0 = 低优先级

图注：			
R = 可读位	W = 可写位	U = 未用位，读为 0	
-n = 上电复位时的值	1 = 置 1	0 = 清零	x = 未知

9.6 RCON 寄存器

RCON 寄存器包含几个标志位，可以用来确定器件上次复位或从空闲或休眠模式被唤醒的原因。RCON 还包含 IPEN 位，该位可以使能中断优先级。

寄存器 9-10： RCON：复位控制寄存器

R/W-0	R/W-1	U-0	R/W-1	R-1	R-1	R/W-0	R/W-0
IPEN	SBOREN	—	$\overline{\text{RI}}$	$\overline{\text{TO}}$	$\overline{\text{PD}}$	$\overline{\text{POR}}$	$\overline{\text{BOR}}$
bit 7							bit 0

bit 7 **IPEN**：中断优先级使能位
1 = 使能中断优先级
0 = 禁止中断优先级 （PIC16CXXX 兼容模式）

bit 6 **SBOREN**：BOR 软件使能位
欲知位操作的详细信息，请参见寄存器 4-1。

bit 5 未用：读为 0

bit 4 **$\overline{\text{RI}}$**：RESET 指令标志位
欲知位操作的详细信息，请参见寄存器 4-1。

bit 3 **$\overline{\text{TO}}$**：看门狗定时器超时溢出标志位
欲知位操作的详细信息，请参见寄存器 4-1。

bit 2 **$\overline{\text{PD}}$**：掉电检测标志位
欲知位操作的详细信息，请参见寄存器 4-1。

bit 1 **$\overline{\text{POR}}$**：上电复位状态位
欲知位操作的详细信息，请参见寄存器 4-1。

bit 0 **$\overline{\text{BOR}}$**：欠压复位状态位
欲知位操作的详细信息，请参见寄存器 4-1。

注： 实际的复位值由器件配置和器件复位的类型决定。更多信息请参见寄存器 4-1。

图注：			
R = 可读位	W = 可写位	U = 未用位，读为 0	
-n = 上电复位时的值	1 = 置 1	0 = 清零	x = 未知

9.7 INTn 引脚中断

RB0/AN12/INT0/FLT0/SDI/SDA、 RB1/AN10/INT1/SCK/SCL 和 RB2/AN8/INT2/VMO 引脚的外部中断是边沿触发的。如果 INTCON2 寄存器中对应的 INTEDGx 位置位 （= 1），则该中断由上升沿触发；如果该位清零，则中断由下降沿触发。当 RBx/INTx 引脚上出现一个有效边沿时，对应的标志位 INTxF 被置 1。通过清零对应的允许位 INTxE，可禁止该中断。在重新允许该中断前，必须在中断服务程序中先用软件将 INTxF 标志位清零。

如果 INTxE 位在进入功耗管理模式前被置 1，则所有的外部中断（INT0、 INT1 和 INT2）均能把处理器从功耗管理模式唤醒。如果全局中断允许位 GIE 被置 1，则处理器将在被唤醒之后跳转到中断向量处执行程序。

INT1 和 INT2 的中断优先级由中断优先级位 INT1IP（INTCON3<6>）和 INT2IP （INTCON3<7>）的值决定。没有与 INT0 相关的优先级位。INT0 始终是一个高优先级的中断源。

9.8 TMR0 中断

在 8 位模式下（默认设置）， TMR0 寄存器的溢出（FFh → 00h）会将标志位 TMR0IF 置 1。在 16 位模式下， TMR0H:TMR0L 寄存器对的溢出（FFFFh → 0000h）将使 TMR0IF 标志位置 1。通过将允许位 TMR0IE（INTCON<5>）置 1 或清零，可以允许 / 禁止该中断。Timer0 的中断优先级由中断优先级位 TMR0IP（INTCON2<2>）的值决定。欲进一步了解 Timer0 模块的详细信息，请参见**第 13.0 节 “Timer2 模块”**。

9.9 PORTB 电平变化中断

PORTB<7:4> 上的输入电平变化会将标志位 RBIF（INTCON<0>）置 1。通过置 1/ 清零允许位 RBIE（INTCON<3>），可以允许 / 禁止该中断。PORTB 电平变化中断的优先级由中断优先级位 RBIP（INTCON2<0>）包含的值决定。

9.10 中断的现场保护

在中断期间，返回的 PC 地址被压入堆栈。另外，WREG、 Status 以及 BSR 寄存器的值被压入快速返回堆栈。如果未使用中断快速返回功能（见**第 5.3 节 “数据存储器构成”**），用户可能需要在进入中断服务程序时，保存 WREG、 Status 以及 BSR 寄存器的值。根据用户的具体应用，可能还需要保存其他寄存器的值。例 9-1 在执行中断服务程序期间，保存并恢复 WREG、Status 和 BSR 寄存器的值。

例 9-1： 将 STATUS、 WREG 和 BSR 寄存器的值保存在 RAM 中

```
MOVWF    W_TEMP                     ; W_TEMP is in virtual bank
MOVFF    STATUS, STATUS_TEMP        ; STATUS_TEMP located anywhere
MOVFF    BSR, BSR_TEMP              ; BSR_TMEP located anywhere
;
; USER ISR CODE
;
MOVFF    BSR_TEMP, BSR              ; Restore BSR
MOVF     W_TEMP, W                  ; Restore WREG
MOVFF    STATUS_TEMP, STATUS        ; Restore STATUS
```

注：

10.0 I/O 端口

根据选定的器件和使能的功能的不同，最多有 5 个端口可供使用。I/O 端口的一些引脚与器件外设功能复用。通常情况下，当使能外设功能时，该引脚就不能作为通用 I/O 引脚使用。

每个端口都有 3 个工作寄存器。它们是：

- TRIS 寄存器 （数据方向寄存器）
- Port 寄存器 （读取器件引脚的电平）
- LAT 寄存器 （输出锁存）

在对 I/O 引脚驱动值进行读—修改—写操作时会用到数据锁存器 （LAT 寄存器）。

图 10-1 给出了不带外设接口的通用 I/O 端口的简化模型。

图 10-1： 通用 I/O 端口的操作

注 1： I/O 引脚与 V_{DD} 和 V_{SS} 之间接有保护二极管。

10.1 PORTA、 TRISA 和 LATA 寄存器

PORTA 是 8 位宽的双向端口。相应的数据方向寄存器为 TRISA。将 TRISA 置 1（= 1）可以将对应的 PORTA 引脚配置为输入引脚（即将对应的输出驱动器置于高阻态）。将 TRISA 位清零 （= 0）可以将对应的 PORTA 引脚配置为输出引脚（即将输出锁存器的内容从选定引脚输出）。

读 PORTA 寄存器读的是引脚的状态，而写该寄存器则会将值写入端口锁存器。

数据锁存器（LATA）也是存储器映射的。对 LATA 寄存器执行读—修改—写操作将读写 PORTA 的输出锁存值。

RA4引脚与Timer0模块的时钟输入功能复用，成为RA4/T0CKI 引脚。RA6 引脚与主振荡器引脚复用；通过在配置寄存器 1H 中选择 （或不选）主振荡器可将该引脚使能为振荡器 （或 I/O）引脚 （详情见**第 25.1 节 “配置位”**）。当不用作端口引脚时，RA6 和与其相关的 TRIS 和 LATA 位读为 0。

RA4 也同 USB 模块复用；它用作外部 USB 收发器的接收输入引脚。要了解 USB 模块的配置详情，请参见**第 17.2 节 “USB 状态和控制”**。

一些 PORTA 引脚与模拟输入、模拟 V_{REF+} 和 V_{REF-} 输入以及比较器参考电压输出复用。通过将 ADCON1 寄存器 （A/D 控制寄存器 1）中的控制位清零或置 1，选择 RA3:RA0 和 RA5 引脚作为 A/D 转换器的输入通道。

> **注：** 在上电复位时，RA5 和 RA3:RA0 被配置为模拟输入并读为 0。 RA4 被配置为数字输入。

而所有其他的 PORTA 引脚都有 TTL 电平输入缓冲器和 CMOS 输出驱动器。

即使在 PORTA 引脚被用作模拟输入的时候，TRISA 寄存器仍然控制 PORTA 引脚的方向。在将它们用作模拟输入时，用户必须确保 TRISA 寄存器中相应的位保持为置 1 状态。

例 10-1： 初始化 PORTA

```
CLRF    PORTA   ; Initialize PORTA by
                ; clearing output
                ; data latches
CLRF    LATA    ; Alternate method
                ; to clear output
                ; data latches
MOVLW   0Fh     ; Configure A/D
MOVWF   ADCON1  ; for digital inputs
MOVLW   07h     ; Configure comparators
MOVWF   CMCON   ; for digital input
MOVLW   0CFh    ; Value used to
                ; initialize data
                ; direction
MOVWF   TRISA   ; Set RA<3:0> as inputs
                ; RA<5:4> as outputs
```

表 **10-1**： **PORTA I/O** 汇总

引脚	功能	TRIS 设置	I/O	I/O 类型	说明
RA0/AN0	RA0	0	OUT	DIG	LATA<0> 数据输出；不受模拟输入影响。
		1	IN	TTL	PORTA<0> 数据输入；使能模拟输入时被禁止。
	AN0	1	IN	ANA	A/D 输入通道 0 和比较器 C1 - 输入。上电复位时默认配置为输入；不影响数字输出。
RA1/AN1	RA1	0	OUT	DIG	LATA<1> 数据输出；不受模拟输入影响。
		1	IN	TTL	PORTA<1> 数据输入；在上电复位时读为 0。
	AN1	1	IN	ANA	A/D 输入通道 1 和比较器 C2 - 输入。上电复位时默认配置为输入；不影响数字输出。
RA2/AN2/VREF-/CVREF	RA2	0	OUT	DIG	LATA<2> 数据输出；不受模拟输入影响。当使能 CVREF 输出时被禁止。
		1	IN	TTL	PORTA<2> 数据输入。当使能模拟功能或 CVREF 输出时被禁止。
	AN2	1	IN	ANA	A/D 输入通道 2 和比较器 C2+ 输入。上电复位时默认配置为输入；不受模拟输出的影响。
	VREF-	1	IN	ANA	A/D 和比较器参考电压 （低）输入。
	CVREF	x	OUT	ANA	比较器参考电压输出。使能此功能将禁止数字 I/O。
RA3/AN3/VREF+	RA3	0	OUT	DIG	LATA<3> 数据输出；不受模拟输入影响。
		1	IN	TTL	PORTA<3> 数据输入；使能模拟输入时被禁止。
	AN3	1	IN	ANA	A/D 输入通道 3 和比较器 C1+ 输入。上电复位时为默认配置。
	VREF+	1	IN	ANA	A/D 和比较器参考电压 （高）输入。
RA4/T0CKI/C1OUT/RCV	RA4	0	OUT	DIG	LATA<4> 数据输出；不受模拟输入影响。
		1	IN	ST	PORTA<4> 数据输入；使能模拟输入时被禁止。
	T0CKI	1	IN	ST	Timer0 时钟输入。
	C1OUT	0	OUT	DIG	比较器 1 输出；优先级高于端口数据。
	RCV	x	IN	TTL	外部 USB 收发器 RCV 输入。
RA5/AN4/$\overline{SS}$/HLVDIN/C2OUT	RA5	0	OUT	DIG	LATA<5> 数据输出；不受模拟输入影响。
		1	IN	TTL	PORTA<5> 数据输入；使能模拟输入时被禁止。
	AN4	1	IN	ANA	A/D 输入通道 4。上电复位时为默认配置。
	$\overline{SS}$	1	IN	TTL	SSP 的从动选择输入 （MSSP 模块）。
	HLVDIN	1	IN	ANA	高 / 低压检测外部跳变点输入。
	C2OUT	0	OUT	DIG	比较器 2 输出；优先级高于端口数据。
OSC2/CLKO/RA6	RA6	0	OUT	DIG	LATA<6> 数据输出。仅在 ECIO、 ECPIO 和 INTIO 模式下可用，否则读为 0。
		1	IN	TTL	PORTA<6> 数据输入。仅在 ECIO、 ECPIO 和 INTIO 模式下可用，否则读为 0。
	OSC2	x	OUT	ANA	主振荡器反馈输出连接 （XT 和 HS 模式）。
	CLKO	x	OUT	DIG	系统周期时钟输出 （FOSC/4）； EC、 ECPLL 和 INTCKO 模式下可用。

图注： OUT = 输出， IN = 输入， ANA = 模拟信号， DIG = 数字输出， ST = 施密特缓冲器输入，
TTL = TTL 缓冲器输入， x = 任意值 （TRIS 位不影响端口方向或者在此可被忽略）。

表 **10-2**： 与 **PORTA** 相关的寄存器汇总

名称	Bit 7	Bit 6	Bit 5	Bit 4	Bit 3	Bit 2	Bit 1	Bit 0	复位值所在页
PORTA	—	RA6[1]	RA5	RA4	RA3	RA2	RA1	RA0	54
LATA	—	LATA6[1]	LATA5	LATA4	LATA3	LATA2	LATA1	LATA0	54
TRISA	—	TRISA6[1]	TRISA5	TRISA4	TRISA3	TRISA2	TRISA1	TRISA0	54
ADCON1	—	—	VCFG1	VCFG0	PCFG3	PCFG2	PCFG1	PCFG0	52
CMCON	C2OUT	C1OUT	C2INV	C1INV	CIS	CM2	CM1	CM0	53
CVRCON	CVREN	CVROE	CVRR	CVRSS	CVR3	CVR2	CVR1	CVR0	53
UCON	—	PPBRST	SE0	PKTDIS	USBEN	RESUME	SUSPND	—	55

图注： — = 未用 （读为 0）。 PORTA 不使用阴影单元。

注 **1**： RA6 及其相关的锁存和数据方向位根据振荡器配置使能为 I/O 引脚；否则，它们读为 0。

10.2 PORTB、 TRISB 和 LATB 寄存器

PORTB 是 8 位宽的双向端口，对应的数据方向寄存器是TRISB。将TRISB位置1（= 1）可以将对应的PORTB引脚配置为输入引脚（即将对应的输出驱动器置于高阻态）。将 TRISB 位清零 （= 0）将把对应的 PORTB 引脚配置为输出引脚（即将输出锁存器的内容从选定引脚输出）。

数据锁存器 （LATB）也是存储器映射的。对 LATB 寄存器进行读—修改—写操作将读写 PORTB 的输出锁存值。

PORTB 的每个引脚都有内部弱上拉电路。单个控制位可以启用所有上拉电路。可以通过将 RBPU 位（INTCON2<7>）清零来启用上拉电路。当 PORTB 端口的引脚配置为输出时，其弱上拉电路会自动切断。此弱上拉电路在上电复位时被禁止。

注：	在上电复位时，默认情况下 RB4:RB0 被配置为模拟输入并读为 0； RB7:RB5 被配置为数字输入。 通过对配置位 PBADEN （CONFIG3H<1>）进行编程，在发生上电复位时还可将 RB4:RB0 配置为数字输入。

4 个 PORTB 引脚 （RB7:RB4）具有电平变化中断功能。只有配置为输入的引脚会导致此类中断发生，当RB7:RB4中的任何一个引脚被配置为输出时，该引脚不再具有电平变化中断功能。当前 RB7:RB4 输入引脚上的电平与 PORTB 上次读入的锁存值进行比较。RB7:RB4 引脚的“不匹配”输出值一起作逻辑或运算，将标志位 RBIF （INTCON<0>）置 1，产生 RB 端口电平变化中断。

电平变化中断可用于唤醒休眠下的器件。用户可用以下方式在中断服务程序中清除该中断：

a) 读或写 PORTB（除了使用 MOVFF（ANY）,PORTB 指令）。这将结束不匹配状态。

b) 将标志位 RBIF 清零。

电平不匹配的状态将会持续地将 RBIF 标志位置 1。而读 PORTB 将结束不匹配状态，并且允许将 RBIF 标志位清零。

建议使用电平变化中断功能实现按键唤醒以及其他仅使用 PORTB 的电平变化中断功能的操作。在使用电平变化中断功能时，建议不要查询 PORTB 的状态。

RB2 和 RB3 引脚与 USB 外设复用，作为外部 USB 收发器的差分信号输出线 （由 TRIS 位设置）。要了解配置 USB 模块与外部收发器工作的更多信息，请参见**第 17.2.2.2 节 “外部收发器”**。

RB4 与 CSSPP 复用，作为并行通信端口 （Streaming Parallel Port，SPP）的片选信号线（由 TRIS 位设置）。在**第 18.0 节 “并行通信端口”**中对该操作进行了详细的讨论。

例 10-2： 初始化 PORTB

```
CLRF    PORTB  ; Initialize PORTB by
               ; clearing output
               ; data latches
CLRF    LATB   ; Alternate method
               ; to clear output
               ; data latches
MOVLW   0Eh    ; Set RB<4:0> as
MOVWF   ADCON1 ; digital I/O pins
               ; (required if config bit
               ; PBADEN is set)
MOVLW   0CFh   ; Value used to
               ; initialize data
               ; direction
MOVWF   TRISB  ; Set RB<3:0> as inputs
               ; RB<5:4> as outputs
               ; RB<7:6> as inputs
```

 初稿

表 **10-3：** **PORTB I/O** 汇总

引脚	功能	TRIS 设置	I/O	I/O 类型	说明
RB0/AN12/INT0/FLT0/SDI/SDA	RB0	0	OUT	DIG	LATB<0> 数据输出；不受模拟输入影响。
		1	IN	TTL	PORTB<0> 数据输入；当 $\overline{RBPU}$ 位清零时启用弱上拉。当使能模拟输入时被禁止 [1]。
	AN12	1	IN	ANA	A/D 输入通道 12[1]。
	INT0	1	IN	ST	外部中断 0 输入。
	FLT0	1	IN	ST	增强型 PWM 故障输入 （ECCP1 模块）；用软件使能。
	SDI	1	IN	ST	SPI 数据输入 （MSSP 模块）。
	SDA	1	OUT	DIG	I^2C™ 数据输出 （MSSP 模块）；优先级高于端口数据。
		1	IN	I^2C/SMB	I^2C 数据输入 （MSSP 模块）；输入类型取决于模块设置。
RB1/AN10/INT1/SCK/SCL	RB1	0	OUT	DIG	LATB<1> 数据输出；不受模拟输入影响。
		1	IN	TTL	PORTB<1> 数据输入；当 $\overline{RBPU}$ 位清零时启用弱上拉。当使能模拟输入时被禁止 [1]。
	AN10	1	IN	ANA	A/D 输入通道 10[1]。
	INT1	1	IN	ST	外部中断 1 输入。
	SCK	0	OUT	DIG	SPI 时钟输出 （MSSP 模块）；优先级高于端口数据。
		1	IN	ST	SPI 时钟输入 （MSSP 模块）。
	SCL	0	OUT	DIG	I^2C 时钟输出 （MSSP 模块）；优先级高于端口数据。
		1	IN	I^2C/SMB	I^2C 时钟输入 （MSSP 模块）；输入类型取决于模块设置。
RB2/AN8/INT2/VMO	RB2	0	OUT	DIG	LATB<2> 数据输出；不受模拟输入影响。
		1	IN	TTL	PORTB<2> 数据输入；当 $\overline{RBPU}$ 位清零时启用弱上拉。当使能模拟输入时被禁止 [1]。
	AN8	1	IN	ANA	A/D 输入通道 8[1]。
	INT2	1	IN	ST	外部中断 2 输入。
	VMO	0	OUT	DIG	外部 USB 收发器 VMO 数据输出。
RB3/AN9/CCP2/VPO	RB3	0	OUT	DIG	LATB<3> 数据输出；不受模拟输入影响。
		1	IN	TTL	PORTB<3> 数据输入；当 $\overline{RBPU}$ 位清零时启用弱上拉。当使能模拟输入时被禁止 [1]。
	AN9	1	IN	ANA	A/D 输入通道 9[1]。
	CCP2[2]	0	OUT	DIG	CCP2 比较和 PWM 输出。
		1	IN	ST	CCP2 捕捉输入。
	VPO	0	OUT	DIG	外部 USB 收发器 VPO 数据输出。
RB4/AN11/KBI0/CSSPP	RB4	0	OUT	DIG	LATB<4> 数据输出；不受模拟输入影响。
		1	IN	TTL	PORTB<4> 数据输入；当 $\overline{RBPU}$ 位清零时启用弱上拉。当使能模拟输入时被禁止 [1]。
	AN11	1	IN	ANA	A/D 输入通道 11[1]。
	KBI0	1	IN	TTL	电平变化中断引脚。
	CSSPP[4]	0	IN	DIG	SPP 片选控制输出。
RB5/KBI1/PGM	RB5	0	OUT	DIG	LATB<5> 数据输出。
		1	IN	TTL	PORTB<5> 数据输入；当 $\overline{RBPU}$ 位清零时启用弱上拉。
	KBI1	1	IN	TTL	电平变化中断引脚。
	PGM	x	IN	ST	单电源编程模式接入 （ICSP™）。由 LVP 配置位使能。 禁止所有其他引脚功能。

图注： OUT = 输出， IN = 输入， ANA = 模拟信号， DIG = 数字输出， ST = 施密特缓冲器输入， I^2C/SMB = I^2C/SM 总线输入缓冲器， TTL = TTL 缓冲器输入， x = 任意值 （TRIS 位不影响端口方向或者在此可被忽略）。

注 1： 上电复位时的配置由 PBADEN 配置位决定。当 PBADEN 置 1 时这些引脚被配置为模拟输入，而当 PBADEN 清零时这些引脚被配置为数字输入。

2： 当 CCP2MX = 0 时，对 CCP2 进行其他分配。默认分配为 RC1。

3： 当使能 ICSP™ 或 ICD 时，禁止所有其他引脚功能。

4： 仅用于 40/44 引脚器件。

表 10-3： PORTB I/O 汇总 （续）

引脚	功能	TRIS 设置	I/O	I/O 类型	说明
RB6/KBI2/PGC	RB6	0	OUT	DIG	LATB<6> 数据输出。
		1	IN	TTL	PORTB<6> 数据输入；当 $\overline{RBPU}$ 位清零时启用弱上拉。
	KBI2	1	IN	TTL	电平变化中断引脚。
	PGC	x	IN	ST	供 ICSP 和 ICD 工作使用的串行执行 （ICSP）时钟输入 (3)。
RB7/KBI3/PGD	RB7	0	OUT	DIG	LATB<7> 数据输出。
		1	IN	TTL	PORTB<7> 数据输入；当 $\overline{RBPU}$ 位清零时启用弱上拉。
	KBI3	1	IN	TTL	电平变化中断引脚。
	PGD	x	OUT	DIG	供 ICSP 和 ICD 工作使用的串行执行数据输出 (3)。
		x	IN	ST	供 ICSP 和 ICD 工作使用的串行执行数据输入 (3)。

图注： OUT = 输出， IN = 输入， ANA = 模拟信号， DIG = 数字输出， ST = 施密特缓冲器输入，
I^2C/SMB = I^2C/SM 总线输入缓冲器， TTL = TTL 缓冲器输入， x = 任意值 （TRIS 位不影响端口方向或者在此可被忽略）。

注 1： 上电复位时的配置由 PBADEN 配置位决定。当 PBADEN 置 1 时这些引脚被配置为模拟输入，而当 PBADEN 清零时这些引脚被配置为数字输入。

2： 当 CCP2MX = 0 时，对 CCP2 进行其他分配。默认分配为 RC1。

3： 当使能 ICSP™ 或 ICD 时，禁止所有其他引脚功能。

4： 仅用于 40/44 引脚器件。

表 10-4： 与 PORTB 相关的寄存器汇总

名称	Bit 7	Bit 6	Bit 5	Bit 4	Bit 3	Bit 2	Bit 1	Bit 0	复位值所在页
PORTB	RB7	RB6	RB5	RB4	RB3	RB2	RB1	RB0	54
LATB	LATB7	LATB6	LATB5	LATB4	LATB3	LATB2	LATB1	LATB0	54
TRISB	TRISB7	TRISB6	TRISB5	TRISB4	TRISB3	TRISB2	TRISB1	TRISB0	54
INTCON	GIE/GIEH	PEIE/GIEL	TMR0IE	INT0IE	RBIE	TMR0IF	INT0IF	RBIF	51
INTCON2	$\overline{RBPU}$	INTEDG0	INTEDG1	INTEDG2	—	TMR0IP	—	RBIP	51
INTCON3	INT2IP	INT1IP	—	INT2IE	INT1IE	—	INT2IF	INT1IF	51
ADCON1	—	—	VCFG1	VCFG0	PCFG3	PCFG2	PCFG1	PCFG0	52
SPPCON(1)	—	—	—	—	—	—	SPPOWN	SPPEN	55
SPPCFG(1)	CLKCFG1	CLKCFG0	CSEN	CLK1EN	WS3	WS2	WS1	WS0	55
UCON	—	PPBRST	SE0	PKTDIS	USBEN	RESUME	SUSPND	—	55

图注： — = 未用 （读为 0）。 PORTB 不使用阴影单元。

注 1： 在 28 引脚器件上这些寄存器不可用。

10.3 PORTC、 TRISC 和 LATC 寄存器

PORTC 是 7 位宽的双向端口，对应的数据方向寄存器是 TRISC。将 TRISC 位置 1 （= 1）可以将对应的 PORTC 引脚设置为输入引脚 （即将对应的输出驱动器置于高阻态） 。将 TRISC 位清零 （= 0）将把对应的 PORTC 引脚设置为输出引脚 （即将输出锁存器的内容从选定的引脚输出）。

在 PIC18F2455/2550/4455/4550 器件中， RC3 引脚不存在。

数据锁存器 （LATC）也是存储器映射的。对 LATC 寄存器执行读—修改—写操作将读写 PORTC 的输出锁存值。

PORTC 主要与串行通信模块复用，包括 EUSART、MSSP模块和USB模块（表 10-5）。除了RC4和RC5，PORTC 使用施密特触发输入缓冲器。

RC4 和 RC5 引脚与 USB 模块复用。根据模块的配置，它们可以作为片内 USB 收发器的差分数据线，或者外部 USB 收发器的数据输入端。RC4 和 RC5 具有 TTL 输入缓冲器，而其他引脚则为施密特触发缓冲器。

与其他 PORTC 引脚不同， RC4 和 RC5 没有与之关联的 TRISC 位。作为数字端口，它们只能用作数字输入。当被配置用于 USB 操作时，给定时间内的数据方向由 USB 模块的配置和状态决定。 如果使用外部收发器，RC4 和 RC5 将始终作为收发器的输入端。 如果使用片内收发器，将由模块当前执行的操作决定数据方向。

当使能外部收发器时，RC1 也用作收发器的输出使能控制线。**第 17.2.2.2 节 “外部收发器”**给出了配置 USB 选项的更多信息。

当使能除 RC4 和 RC5 外的外设功能时，应小心定义每个 PORTC 引脚的 TRIS 位。有些外设会改写 TRIS 位的设置而将引脚定义为输出引脚或输入引脚。用户应该查阅相应的外设章节来正确设置 TRIS 位。

注：	在上电复位时，这些引脚被配置为数字输入。

外设改写会影响TRISC寄存器的内容。尽管外设器件可能会改写一个或多个引脚，读TRISC总是会返回其当前的内容。

例 10-3： 初始化 PORTC

```
CLRF    PORTC   ; Initialize PORTC by
                ; clearing output
                ; data latches
CLRF    LATC    ; Alternate method
                ; to clear output
                ; data latches
MOVLW   07h     ; Value used to
                ; initialize data
                ; direction
MOVWF   TRISC   ; RC<5:0> as outputs
                ; RC<7:6> as inputs
```

表 10-5： PORTC I/O 汇总

引脚	功能	TRIS 设置	I/O	I/O 类型	说明
RC0/T1OSO/T13CKI	RC0	0	OUT	DIG	LATC<0> 数据输出。
		1	IN	ST	PORTC<0> 数据输入。
	T1OSO	x	OUT	ANA	Timer1 振荡器输出；当使能 Timer1 振荡器时被使能。禁止数字 I/O。
	T13CKI	1	IN	ST	Timer1/Timer3 计数器输入。
RC1/T1OSI/CCP2/$\overline{\text{UOE}}$	RC1	0	OUT	DIG	LATC<1> 数据输出。
		1	IN	ST	PORTC<1> 数据输入。
	T1OSI	x	IN	ANA	Timer1 振荡器输入；当使能 Timer1 振荡器时被使能。禁止数字 I/O。
	CCP2[1]	0	OUT	DIG	CCP2 比较和 PWM 输出；优先级高于端口数据。
		1	IN	ST	CCP2 捕捉输入。
	$\overline{\text{UOE}}$	0	OUT	DIG	外部 USB 收发器 $\overline{\text{OE}}$ 输出。
RC2/CCP1/P1A	RC2	0	OUT	DIG	LATC<2> 数据输出。
		1	IN	ST	PORTC<2> 数据输入。
	CCP1	0	OUT	DIG	ECCP1 比较和 PWM 输出；优先级高于端口数据。
		1	IN	ST	ECCP1 捕捉输入。
	P1A[3]	0	OUT	DIG	ECCP1 增强型 PWM 输出，通道 A；优先级高于端口数据。可能在增强型 PWM 关闭事件期间被配置为三态。
RC4/D-/VM	RC4	—[2]	IN	TTL	PORTC<4> 数据输入；使能 USB 时被禁止。
	D-	—[2]	OUT	XCVR	USB 总线差分输出负信号线 （内部收发器）。
		—[2]	IN	XCVR	USB 总线差分输入负信号线 （内部收发器）。
	VM	—[2]	IN	TTL	外部 USB 收发器 VM 输入。
RC5/D+/VP	RC5	—[2]	IN	TTL	PORTC<5> 数据输入；使能 USB 时被禁止。
	D+	—[2]	OUT	XCVR	USB 总线差分输出正信号线 （内部收发器）。
		—[2]	IN	XCVR	USB 总线差分输入正信号线 （内部收发器）。
	VP	—[2]	IN	TTL	外部 USB 收发器 VP 输入。
RC6/TX/CK	RC6	0	OUT	DIG	LATC<6> 数据输出。
		1	IN	ST	PORTC<6> 数据输入。
	TX	0	OUT	DIG	异步串行发送数据输出 （EUSART 模块）；优先级高于端口数据。用户必须将其配置为输出。
	CK	0	OUT	DIG	同步串行时钟输出 （EUSART 模块）；优先级高于端口数据。
		1	IN	ST	同步串行时钟输入 （EUSART 模块）。
RC7/RX/DT/SDO	RC7	0	OUT	DIG	LATC<7> 数据输出。
		1	IN	ST	PORTC<7> 数据输入。
	RX	1	IN	ST	异步串行接收数据输入 （EUSART 模块）。
	DT	1	OUT	DIG	同步串行数据输出 （EUSART 模块）；优先级高于 SPI 和端口数据。
		1	IN	ST	同步串行数据输入 （EUSART 模块）。用户必须将其配置为输入。
	SDO	0	OUT	DIG	SPI 数据输出 （MSSP 模块）；优先级高于端口数据。

图注： OUT = 输出， IN = 输入， ANA = 模拟信号， DIG = 数字输出， ST = 施密特缓冲器输入，
TTL = TTL 缓冲器输入， XCVR = USB 收发器， x = 任意值 （TRIS 位不影响端口方向或者在此可被忽略）。

注 1： 默认引脚分配。当 CCP2MX = 0 时，分配为 RB3。

2： RC4 和 RC5 没有相应的 TRISC 位。在端口模式下，这些引脚只作为输入引脚。USB 配置决定 USB 数据方向。

3： 仅用于 40/44 引脚器件。

 初稿

表 **10-6**： 与 **PORTC** 相关的寄存器汇总

名称	Bit 7	Bit 6	Bit 5	Bit 4	Bit 3	Bit 2	Bit 1	Bit 0	复位值所在页
PORTC	RC7	RC6	RC5[(1)]	RC4[(1)]	—	RC2	RC1	RC0	54
LATC	LATC7	LATC6	—	—	—	LATC2	LATC1	LATC0	54
TRISC	TRISC7	TRISC6	—	—	—	TRISC2	TRISC1	TRISC0	54
UCON	—	PPBRST	SE0	PKTDIS	USBEN	RESUME	SUSPND	—	55

图注： — = 未用 （读为 0）。 PORTC 不使用阴影单元。

注 **1**： 当 USB 模块被禁止时 （UCON<3> = 0）才可将 RC5 和 RC4 用作端口引脚。

10.4 PORTD、 TRISD 和 LATD 寄存器

注：	PORTD 只在 40/44 引脚器件上存在。

PORTD 是 8 位宽的双向端口，对应的数据方向寄存器是 TRISD。将 TRISD 位置 1（= 1）将把对应的 PORTD 引脚配置为输入引脚（即将对应的输出驱动器置于高阻态）。将 TRISD 位清零 （= 0）将把对应的 PORTD 引脚配置为输出引脚（即将输出锁存器的内容从选定的引脚输出）。

数据锁存器 （LATD）也是存储器映射的。对 LATD 寄存器执行读—修改—写操作将读写 PORTD 的输出锁存值。

PORTD 上的所有引脚都配有施密特触发输入缓冲器。每个引脚都可被单独地设置为输入或输出。

PORTD 的每个引脚都有内部弱上拉电路。单个控制位 RDPU（PORTE<7>）可以启用所有上拉电路。通过将 RDPU 置 1 可启用上拉电路。当端口引脚配置为数据输出或与其他外设复用时，其弱上拉电路会自动切断。此弱上拉电路在上电复位时被禁止。**第 10.5节“PORTE、TRISE 和 LATE 寄存器”**将介绍 PORTE 寄存器。

三个 PORTD 引脚与增强型 CCP 模块的输出 P1B、P1C 和 P1D 复用。在**第 16.0 节 “增强型捕捉 / 比较 /PWM（ECCP）模块”**中将对这些额外的 PWM 输出引脚的操作进行详细说明。

注：	在上电复位时，这些引脚被配置为数字输入。

PORTD 也可以被配置为 8 位宽的并行通信端口（SPP）。在这种模式下，输入缓冲器是 TTL 型。如需了解更多有关配置和使用 SPP 的信息，请参见**第 18.0 节 “并行通信端口”**。

注：	当增强型 PWM 模式使用双输出或四输出时， PORTD 的 SSP 功能被自动禁止。

例 10-4： 初始化 PORTD

```
CLRF    PORTD   ; Initialize PORTD by
                ; clearing output
                ; data latches
CLRF    LATD    ; Alternate method
                ; to clear output
                ; data latches
MOVLW   0CFh    ; Value used to
                ; initialize data
                ; direction
MOVWF   TRISD   ; Set RD<3:0> as inputs
                ; RD<5:4> as outputs
                ; RD<7:6> as inputs
```

 初稿

表 10-7： PORTD I/O 汇总

引脚	功能	TRIS 设置	I/O	I/O 类型	说明
RD0/SPP0	RD0	0	OUT	DIG	LATD<0> 数据输出。
		1	IN	ST	PORTD<0> 数据输入。
	SPP0	1	OUT	DIG	SPP<0> 输出数据，优先级高于端口数据。
		1	IN	TTL	SPP<0> 输入数据。
RD1/SPP1	RD1	0	OUT	DIG	LATD<1> 数据输出。
		1	IN	ST	PORTD<1> 数据输入。
	SPP1	1	OUT	DIG	SPP<1> 输出数据，优先级高于端口数据。
		1	IN	TTL	SPP<1> 输入数据。
RD2/SPP2	RD2	0	OUT	DIG	LATD<2> 数据输出。
		1	IN	ST	PORTD<2> 数据输入。
	SPP2	1	OUT	DIG	SPP<2> 输出数据，优先级高于端口数据。
		1	IN	TTL	SPP<2> 输入数据。
RD3/SPP3	RD3	0	OUT	DIG	LATD<3> 数据输出。
		1	IN	ST	PORTD<3> 数据输入。
	SPP3	1	OUT	DIG	SPP<3> 输出数据，优先级高于端口数据。
		1	IN	TTL	SPP<3> 输入数据。
RD4/SPP4	RD4	0	OUT	DIG	LATD<4> 数据输出。
		1	IN	ST	PORTD<4> 数据输入。
	SPP4	1	OUT	DIG	SPP<4> 输出数据，优先级高于端口数据。
		1	IN	TTL	SPP<4> 输入数据。
RD5/SPP5/P1B	RD5	0	OUT	DIG	LATD<5> 数据输出。
		1	IN	ST	PORTD<5> 数据输入
	SPP5	1	OUT	DIG	SPP<5> 输出数据，优先级高于端口数据。
		1	IN	TTL	SPP<5> 输入数据。
	P1B	0	OUT	DIG	ECCP1 增强型 PWM 输出，通道 B，优先级高于端口和 SPP 数据 [(1)]。
RD6/SPP6/P1C	RD6	0	OUT	DIG	LATD<6> 数据输出。
		1	IN	ST	PORTD<6> 数据输入。
	SPP6	1	OUT	DIG	SPP<6> 输出数据，优先级高于端口数据。
		1	IN	TTL	SPP<6> 输入数据。
	P1C	0	OUT	DIG	ECCP1 增强型 PWM 输出，通道 C ；优先级高于端口和 SPP 数据 [(1)]。
RD7/SPP7/P1D	RD7	0	OUT	DIG	LATD<7> 数据输出。
		1	IN	ST	PORTD<7> 数据输入。
	SPP7	1	OUT	DIG	SPP<7> 输出数据，优先级高于端口数据。
		1	IN	TTL	SPP<7> 输入数据。
	P1D	0	OUT	DIG	ECCP1 增强型 PWM 输出，通道 D ；优先级高于端口和 SPP 数据 [(1)]。

图注： OUT = 输出， IN = 输入， ANA = 模拟信号， DIG = 数字输出， ST = 施密特缓冲器输入， TTL = TTL 缓冲器输入。

注 1： 可能在增强型 PWM 关闭事件期间被配置为三态。

表 10-8： 与 PORTD 相关的寄存器汇总

名称	Bit 7	Bit 6	Bit 5	Bit 4	Bit 3	Bit 2	Bit 1	Bit 0	复位值所在页
PORTD(3)	RD7	RD6	RD5	RD4	RD3	RD2	RD1	RD0	54
LATD(3)	LATD7	LATD6	LATD5	LATD4	LATD3	LATD2	LATD1	LATD0	54
TRISD(3)	TRISD7	TRISD6	TRISD5	TRISD4	TRISD3	TRISD2	TRISD1	TRISD0	54
PORTE	RDPU(3)	—	—	—	RE3(1,2)	RE2(3)	RE1(3)	RE0(3)	54
CCP1CON	P1M1(3)	P1M0(3)	DC1B1	DC1B0	CCP1M3	CCP1M2	CCP1M1	CCP1M0	53
SPPCON(3)	—	—	—	—	—	—	SPPOWN	SPPEN	55

图注： — = 未用 （读为 0）。 PORTD 不使用阴影单元。

注 1： 仅当主复位功能被禁止 （MCLRE 配置位 = 0）时使用。

2： RE3 是唯一在 28 引脚和 40/44 引脚器件上都存在的 PORTE 位。其他位只有在 40/44 引脚器件上才存在。

3： 28 引脚器件不使用这些寄存器和 / 或位。

 初稿

10.5 PORTE、 TRISE 和 LATE 寄存器

根据所选择的特定PIC18F2455/2550/4455/4550器件，PORTE 有两种实现方式。

对于 40/44 引脚器件，PORTE 是一个 4 位宽端口。3 个引脚（RE0/AN5/CK1SPP、RE1/AN6/CK2SPP 和 RE2/AN7/OESPP）可被独立配置为输入或输出。这些引脚有施密特触发输入缓冲器。当作为模拟输入时，这些引脚读为 0。

PORTE 对应的数据方向寄存器是 TRISE。将 TRISE 位置 1（= 1）可以将对应的 PORTE 引脚配置为输入引脚（即将对应的输出驱动器置于高阻态）。将 TRISE 位清零（= 0）将把对应的 PORTE 引脚配置为输出引脚（即将输出锁存器的内容从选定引脚输出）。

除了端口数据，PORTE 寄存器（寄存器 10-1）也包含 RDPU 控制位（PORTE<7>），它使能或禁止 PORTD 的弱上拉电路。

即使在 RE 引脚被用作模拟输入的时候，TRISE 寄存器仍然控制其方向。当它们作为模拟输入时，用户必须确保引脚的方向被配置为输入。

> **注：** 在上电复位时，RE2:RE0 被配置为模拟输入。

数据锁存器（LATE）也是存储器映射的。对 LATE 寄存器执行读—修改—写操作将读写 PORTE 的输出锁存值。

PORTE 的第四引脚（MCLR/VPP/RE3）只能作为输入引脚。其操作由 MCLRE 配置位控制。当被配置为端口引脚（MCLRE = 0）时，它只能作为数字输入引脚；因此，其操作与 TRIS 或 LATE 位无关。否则，它充当器件的主复位输入。在任一配置中，RE3 在编程过程中还充当编程电压输入引脚。

> **注：** 在上电复位时，仅当主复位功能被禁止时，RE3 才被使能为数字输入。

例 10-5： 初始化 PORTE

```
CLRF    PORTE   ; Initialize PORTE by
                ; clearing output
                ; data latches
CLRF    LATE    ; Alternate method
                ; to clear output
                ; data latches
MOVLW   0Ah     ; Configure A/D
MOVWF   ADCON1  ; for digital inputs
MOVLW   03h     ; Value used to
                ; initialize data
                ; direction
MOVLW   07h     ; Turn off
MOVWF   CMCON   ; comparators
MOVWF   TRISC   ; Set RE<0> as inputs
                ; RE<1> as outputs
                ; RE<2> as inputs
```

10.5.1 28 引脚器件中的 PORTE

对于 28 引脚器件，PORTE 只有在主复位功能被禁止（MCLRE = 0）时才可用，在上述情形下，PORTE 只有一个位（RE3）可用，并且只能用作输入端口。该引脚的操作如前所述。

寄存器 10-1： PORTE 寄存器

R-0	U-0	U-0	U-0	R/W-1	R/W-1	R/W-1	R/W-1
RDPU[3]	—	—	—	RE3[1,2]	RE2[3]	RE1[3]	RE0[3]
bit 7							bit 0

bit 7 **RDPU：**PORTD 上拉使能位
1 = 按各个端口锁存值使能 PORTD 上拉电路
0 = 禁止 PORTD 所有的上拉电路

bit 6-4 未用：读为 0

bit 3-0 **RE3:RE0：**PORTE 数据输入位[1,2]

注 **1：** 仅当禁止主复位功能时（MCLRE 配置位 = 0）可用，否则读为 0。
2： RE3 是唯一在 28 引脚和 40/44 引脚器件上都实现的 PORTE 位。所以其他位仅在 40/44 引脚器件上才存在。
3： 在 28 引脚器件中未用，读为 0。

图注：			
R = 可读位	W = 可写位	U = 未用位，读为 0	
-n = 上电复位时的值	1 = 置 1	0 = 清零	x = 未知

表 10-9： PORTE I/O 汇总

引脚	功能	TRIS 设置	I/O	I/O 类型	说明
RE0/AN5/CK1SPP	RE0	0	OUT	DIG	LATE<0> 数据输出；不受模拟输入影响。
		1	IN	ST	PORTE<0> 数据输入；使能模拟输入时被禁止。
	AN5	1	IN	ANA	A/D 输入通道 5 ；发生 POR 时为默认配置。
	CK1SPP	0	OUT	DIG	SPP 时钟 1 输出 （使能 SPP）。
RE1/AN6/CK2SPP	RE1	0	OUT	DIG	LATE<1> 数据输出；不受模拟输入影响。
		1	IN	ST	PORTE<1> 数据输入；使能模拟输入时被禁止。
	AN6	1	IN	ANA	A/D 输入通道 6 ；发生 POR 时为默认配置。
	CK2SPP	0	OUT	DIG	SPP 时钟 2 输出 （使能 SPP）。
RE2/AN7/OESPP	RE2	0	OUT	DIG	LATE<2> 数据输出；不受模拟输入影响。
		1	IN	ST	PORTE<2> 数据输入；使能模拟输入时被禁止。
	AN7	1	IN	ANA	A/D 输入通道 7 ；发生 POR 时为默认配置。
	OESPP	0	OUT	DIG	SPP 使能输出 （使能 SPP）。
$\overline{\text{MCLR}}$/$\overline{\text{V}_{PP}}$/RE3	RE3	—[1]	IN	ST	PORTE<3> 数据输入；当 MCLRE 配置位清零时使能。
	$\overline{\text{MCLR}}$	—[1]	IN	ST	外部主复位输入；当 MCLRE 配置位置 1 时使能。
	V_{PP}	—[1]	IN	ANA	高压检测，用于 ICSP™ 模式的接入检测。始终可用，与引脚模式无关。

图注： OUT = 输出， IN = 输入， ANA = 模拟信号， DIG = 数字输出， ST = 施密特缓冲器输入。

注 1： RE3 没有相应的 TRISE<3> 位。该引脚始终为输入，而与模式无关。

表 10-10： 与 PORTE 相关的寄存器汇总

名称	Bit 7	Bit 6	Bit 5	Bit 4	Bit 3	Bit 2	Bit 1	Bit 0	复位值所在页
PORTE	RDPU[3]	—	—	—	RE3[1,2]	RE2[3]	RE1[3]	RE0[3]	54
LATE[3]	—	—	—	—	—	LATE2	LATE1	LATE0	54
TRISE[3]	—	—	—	—	—	TRISE2	TRISE1	TRISE0	54
ADCON1	—	—	VCFG1	VCFG0	PCFG3	PCFG2	PCFG1	PCFG0	52
CMCON	C2OUT	C1OUT	C2INV	C1INV	CIS	CM2	CM1	CM0	53
SPPCON[3]	—	—	—	—	—	—	SPPOWN	SPPEN	55
SPPCFG[3]	CLKCFG1	CLKCFG0	CSEN	CLK1EN	WS3	WS2	WS1	WS0	55

图注： — = 未用 （读为 0）。 PORTE 不使用阴影单元。

注 1： 仅当主复位功能被禁止 （MCLRE 配置位 = 0）时可用。

2： RE3 是唯一在 28 引脚和 40/44 引脚器件上都存在的 PORTE 位。其他位只有在 40/44 引脚器件上才存在。

3： 28 引脚器件不使用这些寄存器和 / 或位。

 初稿

11.0 TIMER0 模块

Timer0 模块具有以下特征：

- 可通过软件选择，作为 8 位或 16 位定时器 / 计数器
- 可读写的寄存器
- 专用的 8 位软件可编程预分频器
- 可选的时钟源 （内部或外部）
- 外部时钟的边沿选择
- 溢出中断

T0CON 寄存器（寄存器 11-1）控制该模块的工作，包括预分频比选择。该寄存器是可读写的。

图 11-1 显示了 8 位模式下 Timer0 模块的简化框图，图 11-2 显示了 16 位模式下 Timer0 模块的简化框图。

寄存器 11-1： T0CON：TIMER0 控制寄存器

R/W-1	R/W-1	R/W-1	R/W-1	R/W-1	R/W-1	R/W-1	R/W-1
TMR0ON	T08BIT	T0CS	T0SE	PSA	T0PS2	T0PS1	T0PS0
bit 7							bit 0

bit 7 **TMR0ON**：Timer0 开关控制位
1 = 使能 Timer0
0 = 禁止 Timer0

bit 6 **T08BIT**：Timer0 8 位 /16 位模式控制位
1 = Timer0 被配置为 8 位定时器 / 计数器
0 = Timer0 被配置为 16 位定时器 / 计数器

bit 5 **T0CS**：Timer0 时钟源选择位
1 = T0CKI 引脚上的传输信号作为时钟源
0 = 内部指令周期时钟 （CLKO）作为时钟源

bit 4 **T0SE**：Timer0 时钟源边沿选择位
1 = 在 T0CKI 引脚上电平的下降沿递增
0 = 在 T0CKI 引脚上电平的上升沿递增

bit 3 **PSA**：Timer0 预分频器分配位
1 = 未给 Timer0 分配预分频器。 Timer0 时钟输入不经过预分频器。
0 = 已给 Timer0 分配了预分频器。 Timer0 时钟输入信号来自预分频器的输出。

bit 2-0 **T0PS2:T0PS0**：Timer0 预分频值选择位
111 = 1:256 预分频值
110 = 1:128 预分频值
101 = 1:64 预分频值
100 = 1:32 预分频值
011 = 1:16 预分频值
010 = 1:8 预分频值
001 = 1:4 预分频值
000 = 1:2 预分频值

图注：			
R = 可读位	W = 可写位	U = 未用位，读为 0	
-n = 上电复位时的值	1 = 置 1	0 = 清零	x = 未知

11.1 Timer0 工作原理

Timer0 既可用作定时器亦可用作计数器；具体的模式由 T0CS 位（T0CON<5>）选择。在定时器模式下，除非选择了不同的预分频值，否则，默认情况下在每个时钟周期该模块都会递增（见**第 11.3 节 "预分频器"**）。如果写入 TMR0 寄存器，那么在随后的两个指令周期内，计数将不再递增。用户可通过将校正值写入 TMR0 寄存器来避开上述问题。

通过将 T0CS 位置 1（= 1）选择计数器模式。在计数器模式下，Timer0 可在 RA4/T0CKI 引脚信号的每个上升沿或下降沿递增。触发递增的边沿由 Timer0 时钟源边沿选择位 T0SE（T0CON<4>）决定。清零此位选择上升沿触发。下面讨论外部时钟输入的限制条件。

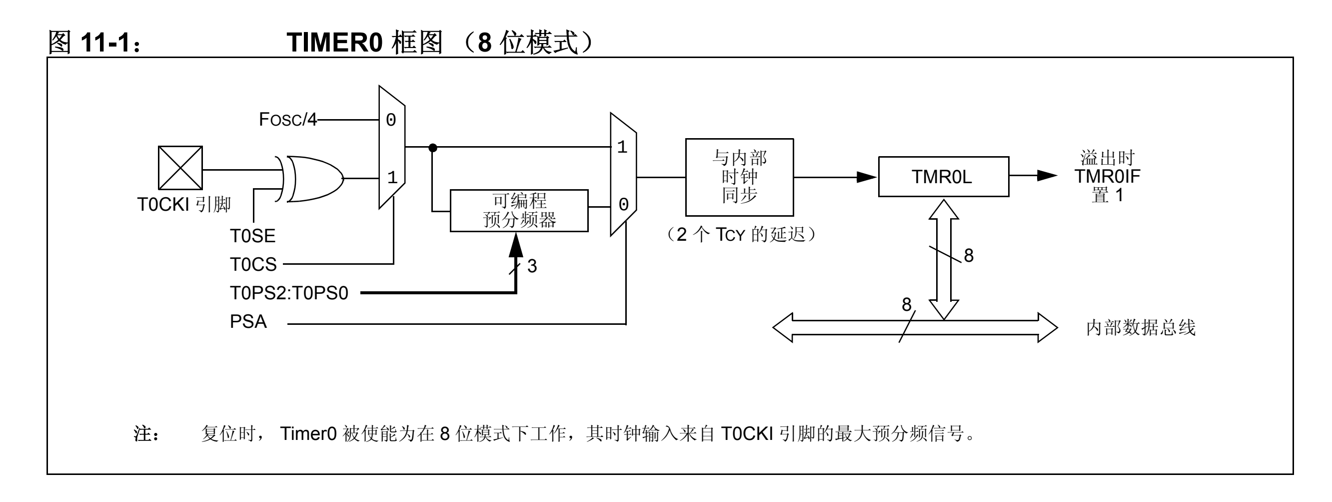
可以使用外部时钟源来驱动 Timer0。但是必须满足一定的要求以确保外部时钟与内部时钟（TOSC）同步。在同步之后，定时器 / 计数器仍需要一定的延时才会引发递增操作。

11.2 Timer0 的 16 位读写模式

TMR0H 不是 16 位模式 Timer0 的高字节，它实际上是 Timer0 高字节的缓冲寄存器， Timer0 的高字节不可以被直接读写（见图 11-2）。在读 TMR0L 时使用 Timer0 高字节的内容更新 TMR0H。这样可以一次读取 Timer0 的全部 16 位，而无需验证读到的高字节和低字节的有效性（在高、低字节分两次连续读取的情况下，由于可能存在进位，因此需要验证读到字节的有效性）。

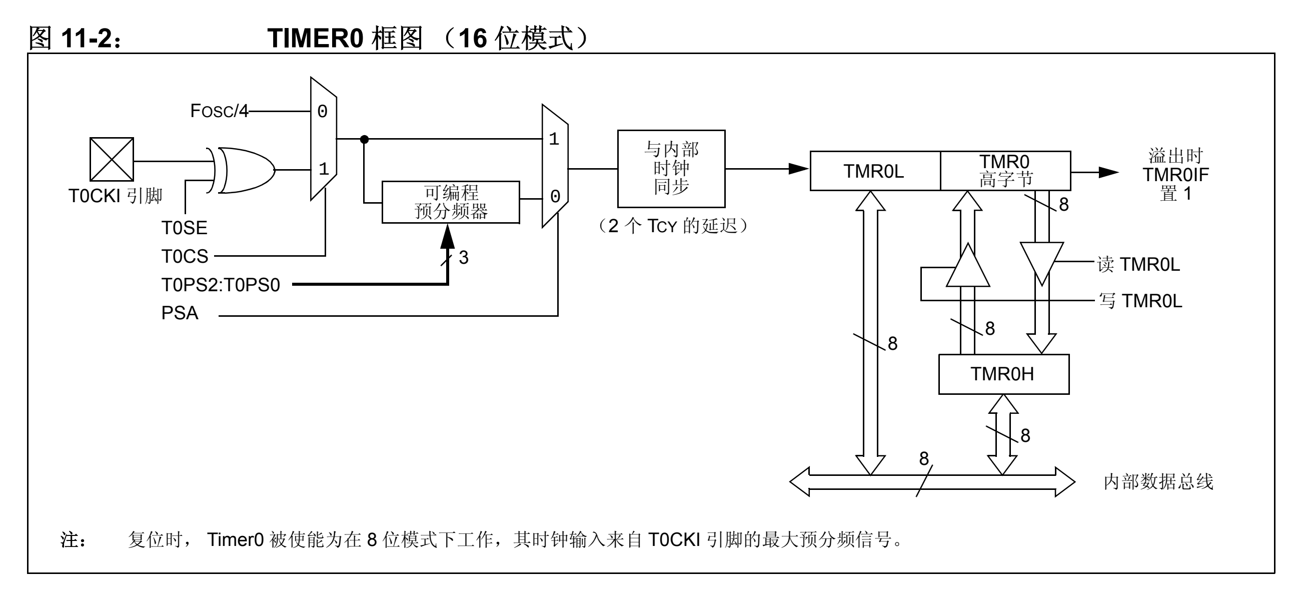
同样，写入 Timer0 的高字节也是通过 TMR0H 缓冲寄存器来操作的。在写入 TMR0L 的同时，使用 TMR0H 的内容更新 Timer0 的高字节。这样一次就可以完成 Timer0 全部 16 位的更新。

图 11-1： TIMER0 框图 （8 位模式）



注： 复位时， Timer0 被使能为在 8 位模式下工作，其时钟输入来自 T0CKI 引脚的最大预分频信号。

图 11-2： TIMER0 框图 （16 位模式）



注： 复位时， Timer0 被使能为在 8 位模式下工作，其时钟输入来自 T0CKI 引脚的最大预分频信号。

11.3 预分频器

Timer0 模块的预分频器为一个 8 位计数器。此预分频器不可直接读写。其值通过 PSA 和 T0PS2:T0PS0 位（T0CON<3:0>）来设置，此位决定预分频器的分配和预分频比值。

将 PSA 位清零可将预分频器分配给 Timer0 模块。预分频比值可以在 1:2 到 1:256 之间进行选择，该比值以 2 的整数次幂递增。

若将预分频器分配给 Timer0 模块，所有以 TMR0 寄存器为写入对象的指令（如 `CLRF TMR0`、`MOVWF TMR0` 和 `BSF TMR0` 等）都将使预分频器的计数值清零。

注：	若将预分频器分配给 Timer0，写入 TMR0 会将预分频器的计数值清零，但不会改变预分频器的分配。

11.3.1 切换预分频器

预分频器的分配完全由软件控制，并且在程序执行期间可以随时更改。

11.4 Timer0 中断

当 TMR0 寄存器发生溢出时（8 位模式下，从 FFh 到 00h；或 16 位模式下，从 FFFFh 到 0000h），将产生 TMR0 中断。这种溢出会将标志位 TMR0IF 置 1。可以通过清零 TMR0IE 位（INTCON<5>）来屏蔽此中断。在重新允许该中断前，必须在中断服务程序中用软件清零 TMR0IF 位。

由于 Timer0 在休眠模式下是关闭的，所以 TMR0 中断无法将处理器从休眠状态唤醒。

表 11-1： 与 TIMER0 相关的寄存器

名称	Bit 7	Bit 6	Bit 5	Bit 4	Bit 3	Bit 2	Bit 1	Bit 0	复位值所在页
TMR0L	Timer0 的低字节寄存器								52
TMR0H	Timer0 的高字节寄存器								52
INTCON	GIE/GIEH	PEIE/GIEL	TMR0IE	INT0IE	RBIE	TMR0IF	INT0IF	RBIF	51
T0CON	TMR0ON	T08BIT	T0CS	T0SE	PSA	T0PS2	T0PS1	T0PS0	52
TRISA	—	TRISA6[1]	TRISA5	TRISA4	TRISA3	TRISA2	TRISA1	TRISA0	54

图注： — = 未用（读为 0）。Timer0 不使用阴影单元。

注 1： 可以根据不同的主振荡器模式将 RA6 配置为端口引脚。被禁止时这些位读为 0。

注：

12.0 TIMER1 模块

Timer1 定时器 / 计数器模块具有以下特征：

- 可通过软件选择，作为 16 位定时器或计数器
- 可读写的 8 位寄存器 （TMR1H 和 TMR1L）
- 可选择使用器件时钟或 Timer1 内部振荡器作为内部或外部时钟源
- 溢出中断
- CCP 特殊事件触发复位
- 器件时钟状态标志位 （T1RUN）

图 12-1 给出了 Timer1 模块的简化框图。图 12-2 给出了此模块在读 / 写模式下的工作原理框图。

该模块自身具有低功耗振荡器，可提供额外的时钟。Timer1 振荡器也可用作单片机处于功耗管理模式下的低功耗时钟源。

以较少的外部元件和代码开销，Timer1 可在应用中提供实时时钟 （Real-Time Clock，RTC）。

Timer1 的工作由 T1CON 控制寄存器 （寄存器 12-1）控制。该寄存器包含 Timer1 振荡器使能位（T1OSCEN）可以通过将控制位 TMR1ON（T1CON<0>）置 1 或清零来使能或禁止 Timer1。

寄存器 12-1： T1CON：TIMER1 控制寄存器

R/W-0	R-0	R/W-0	R/W-0	R/W-0	R/W-0	R/W-0	R/W-0
RD16	T1RUN	T1CKPS1	T1CKPS0	T1OSCEN	$\overline{\text{T1SYNC}}$	TMR1CS	TMR1ON
bit 7							bit 0

bit 7 **RD16**：16 位读 / 写模式使能位
1 = 使能通过一次 16 位操作对 TImer1 寄存器进行读写
0 = 使能通过两次 8 位操作对 Timer1 寄存器进行读写

bit 6 **T1RUN**：Timer1 系统时钟状态位
1 = 器件时钟由 Timer1 振荡器产生
0 = 器件时钟由另一个时钟源产生

bit 5-4 **T1CKPS1:T1CKPS0**：Timer1 输入时钟预分频值选择位
11 = 1:8 预分频值
10 = 1:4 预分频值
01 = 1:2 预分频值
00 = 1:1 预分频值

bit 3 **T1OSCEN**：Timer1 振荡器使能位
1 = 使能 Timer1 振荡器
0 = 关闭 Timer1 振荡器
为了消除功率泄漏，关断了振荡器反相器和反馈电阻 。

bit 2 **$\overline{\text{T1SYNC}}$**：Timer1 外部时钟输入同步选择位
当 TMR1CS = 1 时：
1 = 不与外部时钟输入同步
0 = 与外部时钟输入同步
当 TMR1CS = 0 时 ：
此位被忽略。当 TMR1CS = 0 时， Timer1 使用内部时钟。

bit 1 **TMR1CS**：Timer1 时钟源选择位
1 = 使用 RC0/T1OSO/T13CKI 引脚上的外部时钟 （上升沿触发计数）
0 = 内部时钟 （FOSC/4）

bit 0 **TMR1ON**：Timer1 使能位
1 = 使能 Timer1
0 = 禁止 Timer1

图注：			
R = 可读位	W = 可写位	U = 未用位，读为 0	
-n = 上电复位时的值	1 = 置 1	0 = 清零	x = 未知

12.1 Timer1 工作原理

Timer1 可工作在以下模式：

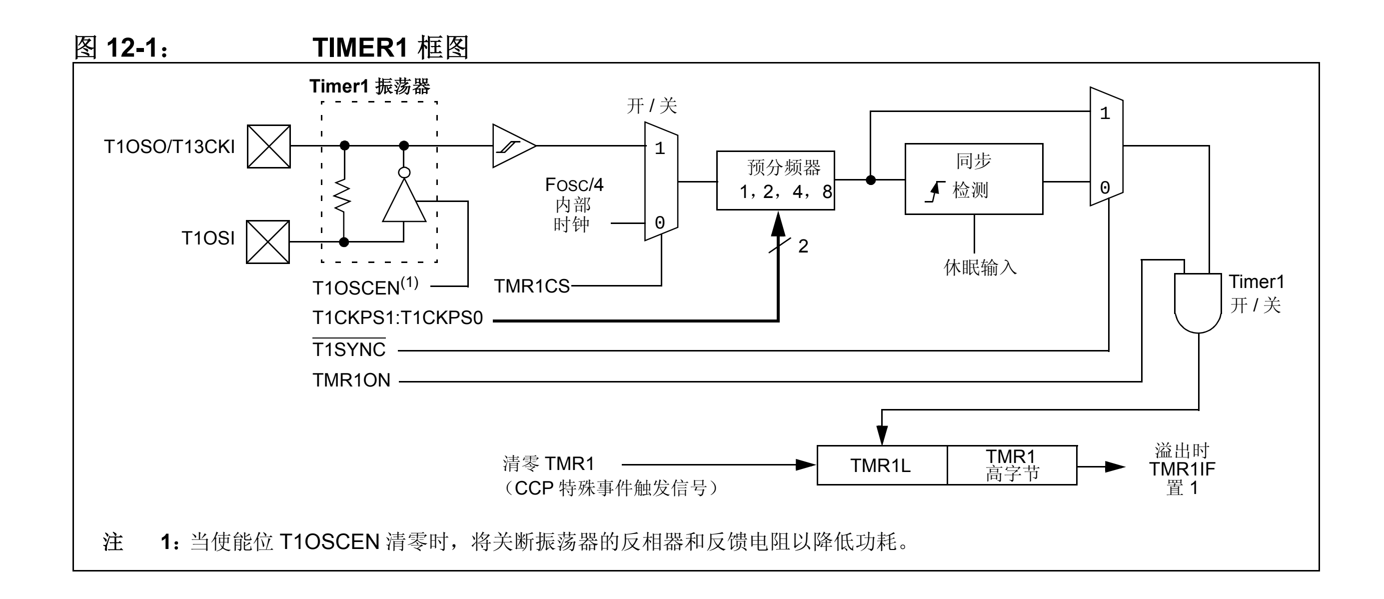
- 定时器
- 同步计数器
- 异步计数器

工作模式由时钟选择位 TMR1CS（T1CON<1>）决定。

当 TMR1CS 清零 （= 0）时， Timer1 在每个内部指令周期（Fosc/4）递增。当 TMR1CS 位置 1 时， Timer1 在 Timer1 外部时钟输入信号或 Timer1 振荡器信号（如果使能）的每个上升沿递增。

当使能 Timer1 时， RC1/T1OSI/$\overline{\text{UOE}}$ 和 RC0/T1OSO/T13CKI 引脚变为输入引脚。这意味着 TRISC<1:0> 的值被忽略并且这些引脚的读取值为 0。

图 12-1： TIMER1 框图



注 1：当使能位 T1OSCEN 清零时，将关断振荡器的反相器和反馈电阻以降低功耗。

图 12-2： TIMER1 框图 （16 位读 / 写模式）

注 1：当使能位 T1OSCEN 清零时，将关闭振荡器的反相器和反馈电阻以降低功耗。

 初稿

12.2 Timer1 16 位读 / 写模式

可将 Timer1 配置为 16 位读写模式 （见图 12-2）。当 RD16 控制位 （T1CON<7>）置 1 时， TMR1H 被映射为 Timer1 的高字节缓冲寄存器。对 TMR1L 的读操作将把 Timer1 的高字节内容装入 Timer1 高字节缓冲器。这种方式使用户可以精确地读取 Time1 的全部 16 位，而不需要像先读高字节再读低字节那样：由于两次读取之间可能存在进位，而不得不验证读取的有效性。

对 Timer1 的高字节进行写操作也必须通过 TMR1H 缓冲器进行。在写入 TMR1L 的同时，使用 TMR1H 的内容更新 Timer1 的高字节。这样允许用户将 16 位值一次写入 Timer1 的高字节和低字节。

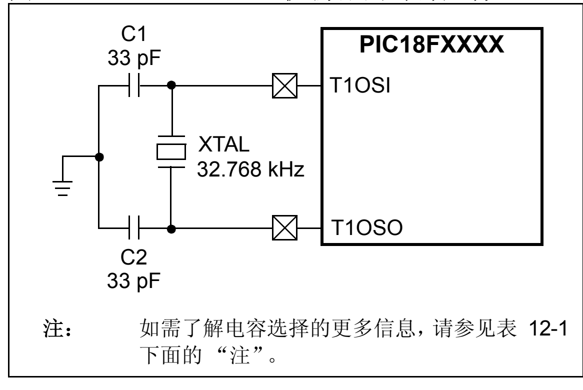
在该模式下不能直接读写 Timer1 的高字节。所有读写都必须通过 Timer1 高字节缓冲器进行。写入 TMR1H 不会清零 Timer1 预分频器。只有在写 TMR1L 时才会清零该预分频器。

12.3 Timer1 振荡器

片上晶体振荡器电路连接在 T1OSI （输入）引脚和 T1OSO （放大器输出）引脚之间。通过将 Timer1 振荡器使能位 T1OSCEN （T1CON<3>）置 1 可使能该振荡器电路。此振荡器是一种低功耗电路，它采用了额定振荡频率为 32 kHz 的晶振，在所有的功耗管理模式下都可继续运行。图 12-3 所示为典型的 LP 振荡电路。表 12-1 给出了供 Timer1 振荡器选择的电容值。

用户必须提供软件延迟来确保 Timer1 振荡器的正常起振。

图 12-3： TIMER1 LP 振荡器的外部元件



注： 如需了解电容选择的更多信息，请参见表 12-1 下面的 “注”。

表 12-1： TIMER 振荡器的电容选择值 [2,3,4]

振荡器类型	频率	C1	C2
LP	32 kHz	27 pF[1]	27 pF[1]

注 1：Microchip 建议在验证振荡电路时从这些值开始。
2：选用较大的电容值虽然可以提高振荡器的稳定性，但同时也会延长起振时间。
3：由于每种谐振器 / 晶振的特性各不相同，因此用户应当向谐振器 / 晶振制造厂商询问外围元件的相应值。
4：上述电容值仅供设计参考。

12.3.1 使用 TIMER1 作为时钟源

在功耗管理模式中也可以将 Timer1 振荡器用作时钟源。通过将时钟选择位 SCS1:SCS0 （OSCCON<1:0>）设置为 01，器件可以切换到 SEC_RUN 模式，此时 CPU 和外设都以 Timer1 振荡器作为时钟源。如果 IDLEN 位（OSCCON<7>）被清零并且执行了 SLEEP 指令，器件将进入 SEC_IDLE 模式。欲知更多详情，请参见**第 3.0 节 “功耗管理模式”**。

只要 Timer1 振荡器提供时钟， Timer1 系统时钟状态标志位 T1RUN （T1CON<6>）就会置 1。这可用于确定控制器的当前时钟模式。该位也可指示故障保护时钟监视器当前正使用的时钟源。如果使能了故障保护时钟监视器并且 Timer1 振荡器在提供时钟信号时发生了故障，查询 T1RUN 位可以确定时钟源是 Timer1 振荡器还是其他时钟源。

12.3.2 低功耗 TIMER1 选项

根据器件配置，Timer1 振荡器可以在两种不同的功耗级别下工作。当 LPT1OSC 配置位置 1 时，Timer1 振荡器在低功耗模式下工作。当 LPT1OSC 清零时，Timer1 在高功耗模式下工作。不管器件工作在什么模式下，特定模式的功耗都是相对固定的。Timer1 的默认配置为工作在功耗较高的模式下。

由于低功耗模式对干扰更加敏感，高噪声环境可能会导致振荡器工作不稳定。因此低功耗模式最适合那些需要重点考虑节省功耗的低噪声应用。

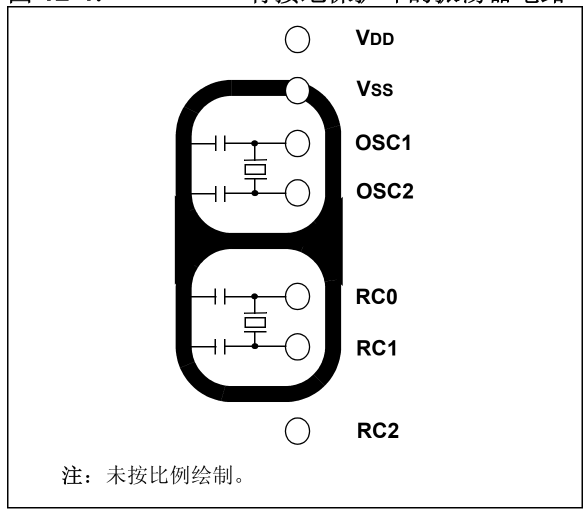
12.3.3 TIMER1 振荡器布线注意事项

Timer1 振荡器在工作期间消耗极少的电流。鉴于此振荡器的低功耗特性，它对附近变化较快的信号可能会比较敏感。

如图 12-3所示，振荡器电路应该尽可能靠近单片机。除了 VSS 或 VDD 外，在振荡电路区域不应有其他电路。

如果必须要在该振荡器附近布置高速电路（如输出比较模式或 PWM 模式下的 CCP1 引脚，或使用 OSC2 引脚的主振荡器），那么在该振荡电路周围布置接地保护环（如图 12-4 所示），对于单面 PCB 板或外加接地层的 PCB 板来讲可能会有帮助。

图 12-4：　有接地保护环的振荡器电路



12.4 Timer1 中断

TMR1 寄存器对（TMR1H:TMR1L）从 0000h 开始递增，一直到 FFFFh，然后溢出从 0000h 重新开始计数。如果允许 Timer1 中断，该中断就会在溢出时产生，并置位中断标志位 TMR1IF（PIR1<0>）。可以通过将 Timer1 中断允许位 TMR1IE（PIE1<0>）置 1 或清零来允许或禁止该中断。

12.5 使用 CCP 特殊事件触发信号来复位 Timer1

如果 CCP 模块被配置为产生特殊事件触发信号的比较模式（CCP1M3:CCP1M0 或 CCP2M3:CCP2M0 = `1011`），该触发信号将复位 Timer1。如果使能了 A/D 模块，来自 CCP2 的触发信号还将启动 A/D 转换（欲知更多信息，请参见**第 15.3.4 节“特殊事件触发器”**）。

要使用这一功能，必须将 Timer1 配置为定时器或同步计数器。在这种情况下，CCPRH:CCPRL 这对寄存器实际上变成了 Timer1 的周期寄存器。

如果 Timer1 工作在异步计数器模式下，复位操作将不起作用。

如果对Timer1的写操作和特殊事件触发信号同时产生，则写操作优先。

注：	来自 CCP2 模块的特殊事件触发信号不会将中断标志位 TMR1IF（PIR1<0>）置 1。

12.6 使用 Timer1 作为实时时钟

为 Timer1 添加外部 LP 振荡器（如**第 12.3 节“Timer1 振荡器”**中所述），可以为用户提供 RTC 功能。这是通过一个提供精确时基的廉价时钟晶振以及几行计算时间的应用程序代码实现的。当器件工作于休眠模式下并使用电池或超大容量电容作为电源时，可以省去额外的 RTC 器件和备用电池。

应用代码程序 `RTCisr`（如例 12-1 所示），给出了使用中断服务程序以 1 秒的间隔递增计数器的简单方法。将 TMR1 寄存器对的值不断加 1 直至溢出，触发中断并调用中断服务程序，该程序会使秒计数器加 1，而分钟和小时计数器则会在前面的计数器溢出时加 1。

由于这对寄存器为 16 位宽，因此使用 32.768 kHz 时钟，将其计数到溢出需要 2 秒。要使溢出按所需的 1 秒间隔进行，必须预先装载这对寄存器。最简单的方法是使用 BSF 指令将 TMR1H 的最高有效位置 1。请注意决不要预先加载或改变 TMR1L 寄存器，这样做可能会引起多个周期的累积错误。

要使此方法精确，Timer1 必须工作在异步模式且必须允许 Timer1 溢出中断（PIE1<0> = `1`），如程序 `RTCinit` 所示。同时 Timer1 振荡器也必须使能并始终保持运行。

例 **12-1：** 使用 **TIMER1** 中断服务实现实时时钟

```
RTCinit
           MOVLW    80h              ; Preload TMR1 register pair
           MOVWF    TMR1H            ; for 1 second overflow
           CLRF     TMR1L
           MOVLW    b'00001111'      ; Configure for external clock,
           MOVWF    T1CON            ; Asynchronous operation, external oscillator
           CLRF     secs             ; Initialize timekeeping registers
           CLRF     mins             ;
           MOVLW    d'12'
           MOVWF    hours
           BSF      PIE1, TMR1IE     ; Enable Timer1 interrupt
           RETURN
RTCisr
           BSF      TMR1H, 7         ; Preload for 1 sec overflow
           BCF      PIR1, TMR1IF     ; Clear interrupt flag
           INCF     secs, F          ; Increment seconds
           MOVLW    d'59'            ; 60 seconds elapsed?
           CPFSGT   secs
           RETURN                    ; No, done
           CLRF     secs             ; Clear seconds
           INCF     mins, F          ; Increment minutes
           MOVLW    d'59'            ; 60 minutes elapsed?
           CPFSGT   mins
           RETURN                    ; No, done
           CLRF     mins             ; clear minutes
           INCF     hours, F         ; Increment hours
           MOVLW    d'23'            ; 24 hours elapsed?
           CPFSGT   hours
           RETURN                    ; No, done
           MOVLW    d'23'            ; 1 hours elapsed?
           MOVWF    hours
           RETURN                    ; Done
```

表 **12-2：** **TIMER1** 作为定时器 **/** 计数器时相关的寄存器

名称	Bit 7	Bit 6	Bit 5	Bit 4	Bit 3	Bit 2	Bit 1	Bit 0	复位值所在页
INTCON	GIE/GIEH	PEIE/GIEL	TMR0IE	INT0IE	RBIE	TMR0IF	INT0IF	RBIF	51
PIR1	SPPIF[(1)]	ADIF	RCIF	TXIF	SSPIF	CCP1IF	TMR2IF	TMR1IF	54
PIE1	SPPIE[(1)]	ADIE	RCIE	TXIE	SSPIE	CCP1IE	TMR2IE	TMR1IE	54
IPR1	SPPIP[(1)]	ADIP	RCIP	TXIP	SSPIP	CCP1IP	TMR2IP	TMR1IP	54
TMR1L	Timer1 寄存器的低字节								52
TMR1H	Timer1 寄存器的高字节								52
T1CON	RD16	T1RUN	T1CKPS1	T1CKPS0	T1OSCEN	$\overline{\text{T1SYNC}}$	TMR1CS	TMR1ON	52

图注： — = 未用 （读为 0）。 Timer1 模块不使用阴影单元。

注 1： 这些位在 28 引脚器件中不使用，始终保持这些位清零。

注：

13.0 TIMER2 模块

Timer2 定时器模块具有以下特征：

- 8 位定时器和周期寄存器 （分别为 TMR2 和 PR2）
- 可读写 （以上两个寄存器）
- 可软件编程的预分频器 （分频比为 1:1、 1:4 和 1:16）
- 可软件编程的后分频器 （分频比为 1:1 至 1:16）
- 当 TMR2 与 PR2 匹配时产生中断
- 作为 MSSP 模块的可选移位时钟

此模块的工作由 T2CON 寄存器（寄存器 13-1）控制，此寄存器用于使能或禁止定时器并配置预分频器和后分频器。可以通过清零控制位 TMR2ON （T2CON<2>）关闭 Timer2，以进一步降低功耗。

图 13-1 所示为该模块的简化框图。

13.1 Timer2 工作原理

在正常操作中， TMR2 从 00h 开始，每个时钟周期（Fosc/4）计数一次。2 位计数器 / 预分频器提供了对时钟输入不分频、4 分频和 16 分频三种预分频选项。可通过预分频控制位 T2CKPS1:T2CKPS0 （T2CON<1:0>）选择预分频值。在每个时钟周期， TMR2 的值都会与周期寄存器 PR2 中的值进行比较。当两个值匹配时，由比较器产生匹配信号作为定时器的输出。此信号也会使 TMR2 的值在下一个周期复位到 00h，并驱动输出计数器 / 后分频器 （见第 **13.2 节 “Timer2 中断”**）。

TMR2 和 PR2 寄存器均可直接读写。在器件复位时，TMR2寄存器都会清零，而PR2寄存器则初始化为FFh。发生以下事件时，预分频和后分频计数器均会清零：

- 对 TMR2 寄存器进行写操作
- 对 T2CON 寄存器进行写操作
- 任何方式的器件复位 （上电复位、 $\overline{\text{MCLR}}$ 复位、看门狗定时器复位或者欠压复位）

写 T2CON 时 TMR2 不会清零。

寄存器 13-1： T2CON：TIMER2 控制寄存器

U-0	R/W-0	R/W-0	R/W-0	R/W-0	R/W-0	R/W-0	R/W-0
—	T2OUTPS3	T2OUTPS2	T2OUTPS1	T2OUTPS0	TMR2ON	T2CKPS1	T2CKPS0
bit 7							bit 0

bit 7 未用：读为 `0`

bit 6-3 **T2OUTPS3:T2OUTPS0**：Timer2 输出后分频比选择位
`0000` = 1:1 后分频比
`0001` = 1:2 后分频比
•
•
•
`1111` = 1:16 后分频比

bit 2 **TMR2ON**：Timer2 使能位
`1` = 使能 Timer2
`0` = 禁止 Timer2

bit 1-0 **T2CKPS1:T2CKPS0**：Timer2 时钟预分频值选择位
`00` = 预分频值为 1
`01` = 预分频值为 4
`1x` = 预分频值为 16

图注：			
R = 可读位	W = 可写位	U = 未用位，读为 0	
-n = 上电复位时的值	1 = 置 1	0 = 清零	x = 未知

13.2 Timer2 中断

Timer2 也可以产生可选的器件中断。 Timer2 输出信号（TMR2 和 PR2 匹配时）可作为 4 位输出计数器 / 后分频器的输入信号。此计数器产生的 TMR2 匹配中断，由其标志位 TMR2IF（PIR1<1>）表示。可以通过将 TMR2 匹配中断允许位 TMR2IE （PIE1<1>）置 1 来允许此中断。

可以通过后分频比控制位 T2OUTPS3:T2OUTPS0（T2CON<6:3>）在 16 个后分频比选项（从 1:1 到 1:16）中进行选择。

13.3 TMR2 输出

TMR2 的不经分频的输出主要用于 CCP 模块，它用作 CCP 模块在 PWM 模式下工作时的时基。

还可将 Timer2 用作 MSSP 模块在 SPI 模式下工作时的移位时钟源。**第 19.0 节 “主控同步串口 （MSSP）模块”**提供了更多信息。



图 13-1： TIMER2 框图

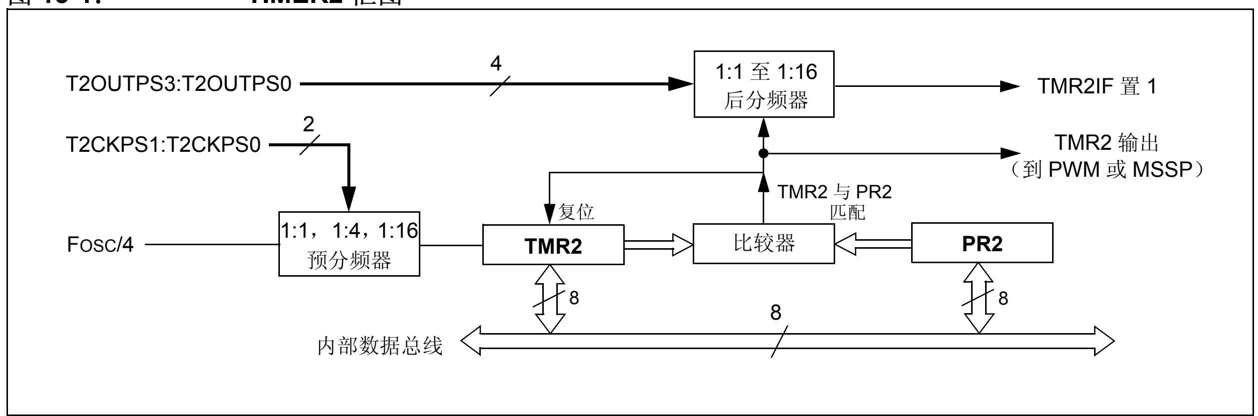


表 13-1： TIMER2 作为定时器 / 计数器时相关的寄存器

名称	Bit 7	Bit 6	Bit 5	Bit 4	Bit 3	Bit 2	Bit 1	Bit 0	复位值所在页
INTCON	GIE/GIEH	PEIE/GIEL	TMR0IE	INT0IE	RBIE	TMR0IF	INT0IF	RBIF	51
PIR1	SPPIF[(1)]	ADIF	RCIF	TXIF	SSPIF	CCP1IF	TMR2IF	TMR1IF	54
PIE1	SPPIE[(1)]	ADIE	RCIE	TXIE	SSPIE	CCP1IE	TMR2IE	TMR1IE	54
IPR1	SPPIP[(1)]	ADIP	RCIP	TXIP	SSPIP	CCP1IP	TMR2IP	TMR1IP	54
TMR2	Timer2 寄存器								52
T2CON	—	T2OUTPS3	T2OUTPS2	T2OUTPS1	T2OUTPS0	TMR2ON	T2CKPS1	T2CKPS0	52
PR2	Timer2 周期寄存器								52

图注： — = 未用 （读为 0）。 Timer2 模块不使用阴影单元。

注 1： 这些位在 28 引脚器件中不使用，始终保持这些位清零。

 初稿

14.0 TIMER3 模块

Timer3 定时器 / 计数器模块具有以下特征：

- 可通过软件选择，作为 16 位定时器或计数器
- 可读写的 8 位寄存器 （TMR3H 和 TMR3L）
- 可选择使用器件时钟或 Timer1 内部振荡器作为内部或外部时钟源
- 溢出中断
- CCP 特殊事件触发模块复位

图 14-1 所示为 Timer3 模块的简化框图。图 14-2 所示为此模块在读 / 写模式下的工作原理框图。

Timer3 模块由 T3CON 寄存器 （寄存器 14-1）控制。该控制寄存器还用于为 CCP 模块选择时钟源 （欲知更多信息，请参见**第 15.1.1 节 “CCP 模块和定时器资源”**）。

寄存器 14-1： T3CON：TIMER3 控制寄存器

R/W-0	R/W-0	R/W-0	R/W-0	R/W-0	R/W-0	R/W-0	R/W-0
RD16	T3CCP2	T3CKPS1	T3CKPS0	T3CCP1	$\overline{\text{T3SYNC}}$	TMR3CS	TMR3ON
bit 7							bit 0

bit 7 **RD16**：16 位读 / 写模式使能位
1 = 使能通过一次 16 位操作对 Timer3 寄存器进行读写
0 = 使能通过两次 8 位操作对 Timer3 寄存器进行读写

bit 6, 3 **T3CCP2:T3CCP1**：CCPx 的时钟源 （是 Timer3 还是 Timer1）使能位
1x = Timer3 是两个 CCP 模块的捕捉 / 比较时钟源
01 = Timer3 是 CCP2 模块的捕捉 / 比较时钟源
Timer1 是 CCP1 模块的捕捉 / 比较时钟源
00 = Timer1 是两个 CCP 模块的捕捉 / 比较时钟源

bit 5-4 **T3CKPS1:T3CKPS0**：Timer3 输入时钟预分频值选择位
11 = 1:8 预分频值
10 = 1:4 预分频值
01 = 1:2 预分频值
00 = 1:1 预分频值

bit 2 **$\overline{\text{T3SYNC}}$**：Timer3 外部时钟输入同步控制位
（不适用于器件时钟来自 Timer1/Timer3。）
<u>当 TMR3CS = 1 时：</u>
1 = 不与外部时钟输入同步
0 = 与外部时钟输入同步
<u>当 TMR3CS = 0 时：</u>
此位被忽略。当 TMR3CS = 0 时， Timer3 使用内部时钟。

bit 1 **TMR3CS**：Timer3 时钟源选择位
1 = 使用 Timer1 振荡器或 T13CKI 引脚信号作为外部时钟输入 （在第一个下降沿之后的上升沿开始计数）
0 = 内部时钟 （FOSC/4）

bit 0 **TMR3ON**：Timer3 使能位
1 = 使能 Timer3
0 = 禁止 Timer3

图注：			
R = 可读位	W = 可写位	U = 未用位，读为 0	
-n = 上电复位时的值	1 = 置 1	0 = 清零	x = 未知

14.1 Timer3 工作原理

Timer3 有三种工作模式：

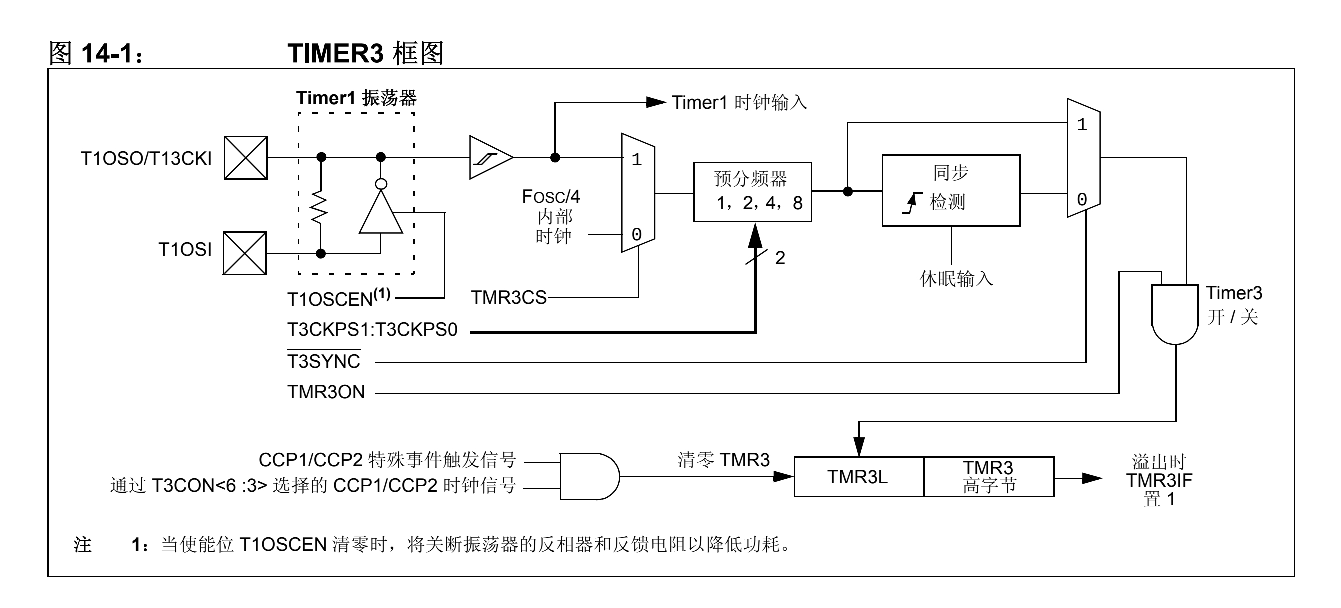
- 定时器
- 同步计数器
- 异步计数器

工作模式由时钟选择位 TMR3CS（T3CON<1>）决定。

当 TMR3CS 清零 （= 0）时，Timer3 在每个内部指令周期 （Fosc/4）递增。当 TMR3CS 置 1 时， Timer3 在 Timer1 外部时钟输入信号或 Timer1 振荡器（如果使能）输出信号的每个上升沿递增。

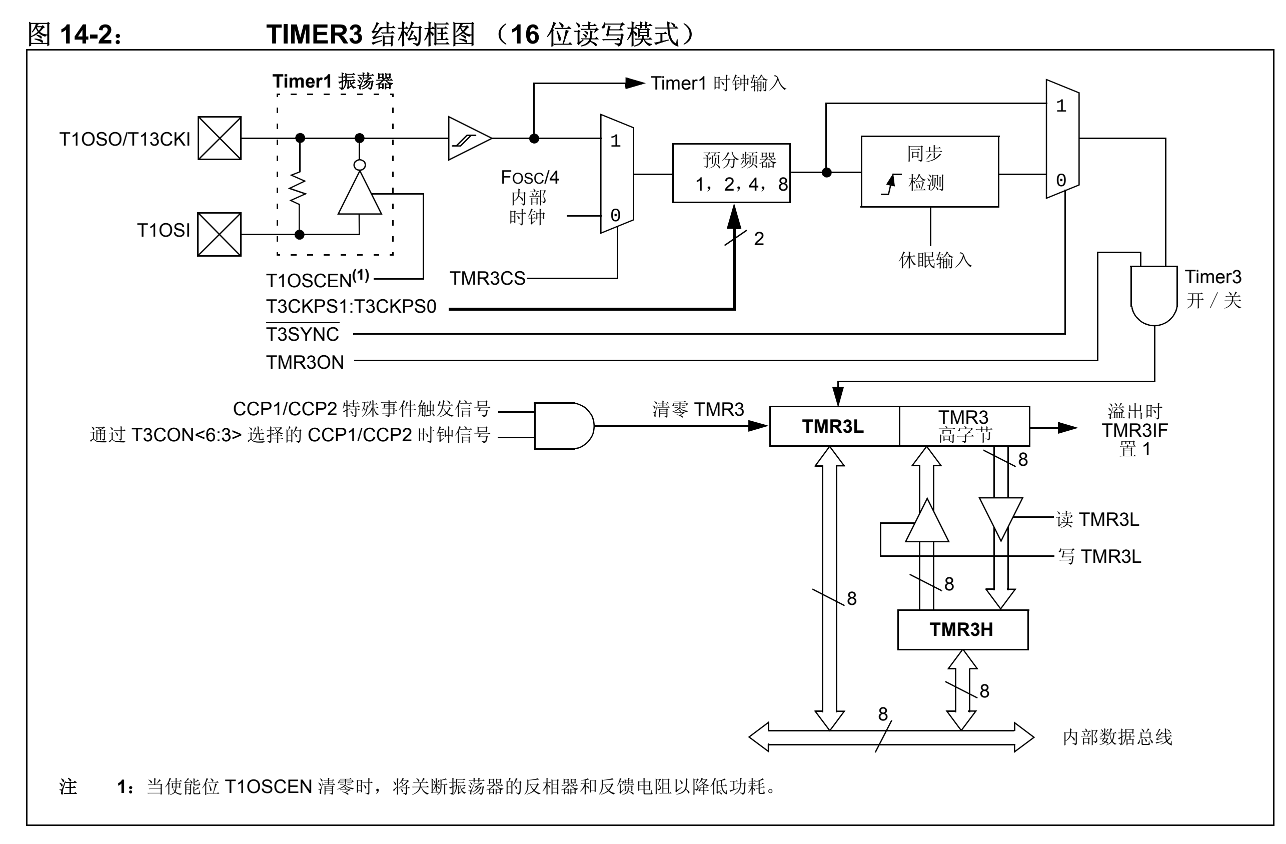
当使能 Timer1 振荡器时，RC1/T1OSI/$\overline{\text{UOE}}$ 和 RC0/T1OSO/T13CKI 引脚变为输入引脚。这意味着 TRISC<1:0> 的值被忽略并且这些引脚的读取值为 0。

图 14-1： TIMER3 框图



注 1：当使能位 T1OSCEN 清零时，将关断振荡器的反相器和反馈电阻以降低功耗。

图 14-2： TIMER3 结构框图 （16 位读写模式）



注 1：当使能位 T1OSCEN 清零时，将关断振荡器的反相器和反馈电阻以降低功耗。

14.2 Timer3 16 位读 / 写模式

可将 Timer3 配置为 16 位读写模式 （见图 14-2）。当 RD16 控制位 （T3CON<7>）置 1 时， TMR3H 的地址被映射到 Timer3 的高字节缓冲寄存器。对 TMR3L 的读操作将把 Timer3 的高位字节内容装入 Timer3 高字节缓冲寄存器。这种方式使用户可以精确地读取 Time3 的全部 16 位，而不需要像先读高字节再读低字节那样：由于两次读取之间可能存在进位，而不得不验证读取的有效性。

对 Timer3 的高字节进行写操作也必须通过 TMR3H 缓冲器进行。在写入 TMR3L 的同时，使用 TMR3H 的内容更新 Timer3 的高字节。这样允许用户将所有的 16 位值一次写入 Timer3 的高字节和低字节。

在该模式下不能直接读写 Timer3 的高字节。所有读写都必须通过 Timer3 高字节缓冲器进行。

写入 TMR3H 不会清零 Timer3 预分频器。只有在写入 TMR3L 时才会清零该预分频器。

14.3 使用 Timer1 振荡器作为 Timer3 的时钟源

Timer1 内部振荡器可用作 Timer3 的时钟源。通过将 T1OSCEN（T1CON<3>）位置 1，可使能 Timer1 振荡器。要将它用作 Timer3 的时钟源，还必须将 TMR3CS 位置 1。如前所述，这样做也会将 Timer3 配置为在振荡器时钟源的每个上升沿递增。

在第 **12.0 节 "Timer1 模块"**中对 Timer1 振荡器做出了说明。

14.4 Timer3 中断

TMR3 寄存器对（TMR3H:TMR3L）从 0000h 开始递增直到 FFFFh，然后溢出返回 0000h。如果允许 Timer3 中断，该中断就会在溢出时产生，并置位中断标志位 TMR3IF （PIR2<1>）。可以通过对 Timer3 中断允许位 TMR3IE（PIE2<1>）置 1 或清零来允许或禁止该中断。

14.5 使用 CCP 特殊事件触发信号来复位 Timer3

如果 CCP2 模块被配置为产生特殊事件触发信号的比较模式 （CCP2M3:CCP2M0 = `1011`），该触发信号将复位 Timer3。如果使能了 A/D 模块，还将启动 A/D 转换（欲知更多信息，请参见**第 15.3.4 节 "特殊事件触发器"**。）。

要使用这一功能，必须将 Timer3 配置为定时器或同步计数器。在这种情况下， CCPR2H:CCPR2L 这对寄存器实际上变成了 Timer3 的周期寄存器。

如果 Timer3 工作在异步计数器模式下，复位操作将不起作用。

如果对Timer3的写操作和特殊事件触发信号同时产生，则写操作优先。

注：	来自 CCP2 模块的特殊事件触发信号不会将中断标志位 TMR3IF（PIR2<1>）置 1。

表 14-1： TIMER3 作为定时器 / 计数器时相关的寄存器

名称	Bit 7	Bit 6	Bit 5	Bit 4	Bit 3	Bit 2	Bit 1	Bit 0	复位值所在页
INTCON	GIE/GIEH	PEIE/GIEL	TMR0IE	INT0IE	RBIE	TMR0IF	INT0IF	RBIF	51
PIR2	OSCFIF	CMIF	USBIF	EEIF	BCLIF	HLVDIF	TMR3IF	CCP2IF	54
PIE2	OSCFIE	CMIE	USBIE	EEIE	BCLIE	HLVDIE	TMR3IE	CCP2IE	54
IPR2	OSCFIP	CMIP	USBIP	EEIP	BCLIP	HLVDIP	TMR3IP	CCP2IP	54
TMR3L	Timer3 寄存器的低字节								53
TMR3H	Timer3 寄存器的高字节								53
T1CON	RD16	T1RUN	T1CKPS1	T1CKPS0	T1OSCEN	$\overline{\text{T1SYNC}}$	TMR1CS	TMR1ON	52
T3CON	RD16	T3CCP2	T3CKPS1	T3CKPS0	T3CCP1	$\overline{\text{T3SYNC}}$	TMR3CS	TMR3ON	53

图注： — = 未用 （读为 `0`）。 Timer3 模块不使用阴影单元。

注：

15.0 捕捉 / 比较 /PWM （CCP）模块

PIC18F2455/2550/4455/4550 器件配有两个 CCP （捕捉 / 比较 /PWM）模块。每个模块包含一个 16 位寄存器，它可以用作 16 位捕捉寄存器、16 位比较寄存器或 PWM 主 / 从占空比寄存器。

在28引脚器件中，两个标准CCP模块（CCP1和CCP2）以本章中描述的方式工作。在 40/44 引脚器件中，CCP1 是增强型 CCP 模块，具有标准捕捉和比较模式以及增强型 PWM 模式。在第 **16.0 节 “增强型捕捉 / 比较 /PWM （ECCP）模块”**中将讨论 ECCP 的使用。

在本章中描述的捕捉和比较操作适用于所有标准和增强型 CCP 模块。

> **注：** 在本章和第 **16.0 节“增强型捕捉 / 比较 /PWM （ECCP）模块”**中，在提到 CCP 模块的寄存器和位名称时，一般会使用 “x” 或 “y” 代替特定的模块编号。因此，“CCPxCON” 可能指 CCP1、CCP2 或 ECCP1 的控制寄存器。在这些章节中 “CCPxCON” 用来指模块控制寄存器，无论CCP模块是标准型还是增强型。

寄存器 15-1： CCPxCON：标准 CCP 控制寄存器

U-0	U-0	R/W-0	R/W-0	R/W-0	R/W-0	R/W-0	R/W-0
—(1)	—(1)	DCxB1	DCxB0	CCPxM3	CCPxM2	CCPxM1	CCPxM0
bit 7							bit 0

bit 7-6 **未用：**读为 `0`

注 1：在 28 引脚器件中这些位未用，读为 `0`。

bit 5-4 **DCxB1:DCxB0：**CCPx 模块的 PWM 占空比 bit 1 和 bit 0

捕捉模式：
未用。

比较模式：
未用。

PWM 模式：
这两位是 10 位 PWM 占空比的低 2 位 （bit 1 和 bit 0）。占空比的高 8 位 （DCx9:DCx2）在 CCPRxL 中。

bit 3-0 **CCPxM3:CCPxM0：**CCPx 模块模式选择位

`0000` = 禁止捕捉 / 比较 /PWM （复位 CCP 模块）
`0001` = 保留
`0010` = 比较模式，匹配时输出电平翻转 （CCPIF 位置 1）
`0011` = 保留
`0100` = 捕捉模式，在每个下降沿发生捕捉
`0101` = 捕捉模式，在每个上升沿发生捕捉
`0110` = 捕捉模式，每 4 个上升沿发生捕捉
`0111` = 捕捉模式，每 16 个上升沿发生捕捉
`1000` = 比较模式：CCP 引脚初始化为低电平，比较匹配时，强制 CCP 引脚为高电平 （CCPIF 位置 1）
`1001` = 比较模式：CCP 引脚初始化为高电平，比较匹配时，强制 CCP 引脚为低电平 （CCPIF 位置 1）
`1010` = 比较模式：比较匹配时产生软件中断 （CCPIF 位置 1， CCP 引脚反映出 I/O 状态）
`1011` = 比较模式：CCP2 匹配时产生特殊事件触发信号，复位定时器并启动 A/D 转换 （CCPIF 位置 1）
`11xx` = PWM 模式

图注：			
R = 可读位	W = 可写位	U = 未用位，读为 0	
-n = 上电复位时的值	1 = 置 1	0 = 清零	x = 未知

15.1 CCP 模块配置

每个捕捉 / 比较 /PWM 模块均与一个控制寄存器（通常为 CCPxCON）和一个数据寄存器（CCPRx）相对应。数据寄存器由 2 个 8 位寄存器组成：CCPRxL （低字节）和CCPRxH（高字节）。所有寄存器都是可读写的。

15.1.1 CCP 模块和定时器资源

CCP 模块依据选定的工作模式使用 Timer1、Timer2 或 Timer3。Timer1 和 Timer3 适用于工作在捕捉或比较模式下的模块，而 Timer2 适用于工作在 PWM 模式下的模块。

表 15-1： CCP 模式——定时器资源

CCP/ECCP 模式	定时器资源
捕捉	Timer1 或 Timer3
比较	Timer1 或 Timer3
PWM	Timer2

要将哪个特定的定时器分配给 CCP 模块是由 T3CON 寄存器 （寄存器 14-1）中 “将定时器分配给 CCP 模块”使能位决定的。如果将两个模块配置为同时在相同的模式（捕捉 / 比较或 PWM）下工作，那么这两个模块在某一时刻可能均处于激活状态，并可共用相同的定时器资源。图 15-2 概括了两个模块之间的相互关系。处于异步计数器模式的 Timer1，将无法进行捕捉操作。

15.1.2 CCP2 引脚分配

可根据器件配置改变 CCP2 的引脚分配（捕捉输入、比较和 PWM 输出）。 CCP2MX 配置位决定哪个引脚与 CCP2 复用。默认情况下， CCP2 引脚被分配给 RC1 （CCP2MX = 1）。如果该配置位清零，则 CCP2 与 RB3 复用。

改变 CCP2 的引脚分配不会自动更改端口引脚的配置。用户必须始终保证与 CCP2 操作相对应的 TRIS 寄存器已被正确配置。

表 15-2： CCP1 和 CCP2 在使用定时器资源方面的相互关系

CCP1 模式	CCP2 模式	相互关系
捕捉	捕捉	每个模块都可以使用 TMR1 或 TMR3 作为时基。每个 CCP 的时基可以不同。
捕捉	比较	可将 CCP2 配置为特殊事件触发器以复位 TMR1 或 TMR3 （取决于所使用的时基）。也可以将其用于自动触发 A/D 转换。如果 CCP1 使用同一个定时器作为时基， CCP1 的操作将会受到影响。
比较	捕捉	可将 CCP1 配置为特殊事件触发器以复位 TMR1 或 TMR3 （取决于所使用的时基）。如果 CCP2 使用同一个定时器作为时基， CCP2 的操作将会受到影响。
比较	比较	两个模块都可被配置为特殊事件触发器以复位时基。 CCP2 还可自动触发 A/D 转换。如果两个模块使用同一个时基，将会发生冲突。
捕捉	PWM[1]	无
比较	PWM[1]	无
PWM[1]	捕捉	无
PWM[1]	比较	无
PWM[1]	PWM	两个 PWM 将具有相同的频率和更新速率 （TMR2 中断）。

注 1： 包括标准型和增强型 PWM 操作。

15.2 捕捉模式

在捕捉模式下，当对应的 CCPx 引脚上有事件发生时，CCPRxH:CCPRxL 寄存器对将捕捉 TMR1 或 TMR3 寄存器的 16 位值。事件定义如下：

- 每个下降沿
- 每个上升沿
- 每 4 个上升沿
- 每 16 个上升沿

通过模式选择位 CCPxM3:CCPxM0（CCPxCON<3:0>）选择事件类型。当一个捕捉发生时，中断请求标志位 CCPxIF 置 1，它必须用软件清零。如果在寄存器 CCPRx 中的值被读取之前发生了另一次捕捉，那么之前捕捉的值将被新捕捉的值覆盖。

15.2.1 CCP 引脚配置

在捕捉模式下，应该通过将对应的 TRIS 方向位置 1 来将相应的 CCPx 引脚配置为输入。

注：	如果 RB3/CCP2 或 RC1/CCP2 引脚被配置为输出，对该端口的写操作会产生捕捉条件。

15.2.2 TIMER1/TIMER3 模式选择

用于捕捉的定时器（Timer1 和 / 或 Timer3）必须运行在定时器模式或同步计数器模式下。在异步计数器模式下，无法进行捕捉操作。用于每个 CCP 模块的定时器由 T3CON 寄存器选择（见第 **15.1.1 节“CCP 模块和定时器资源”**）。

15.2.3 软件中断

当捕捉模式改变时，可能会产生错误捕捉中断。用户应该保持 CCPxIE 中断允许位清零，以避免错误中断。还应该在工作模式发生任何改变之后清零中断标志位 CCPxIF。

15.2.4 CCP 预分频器

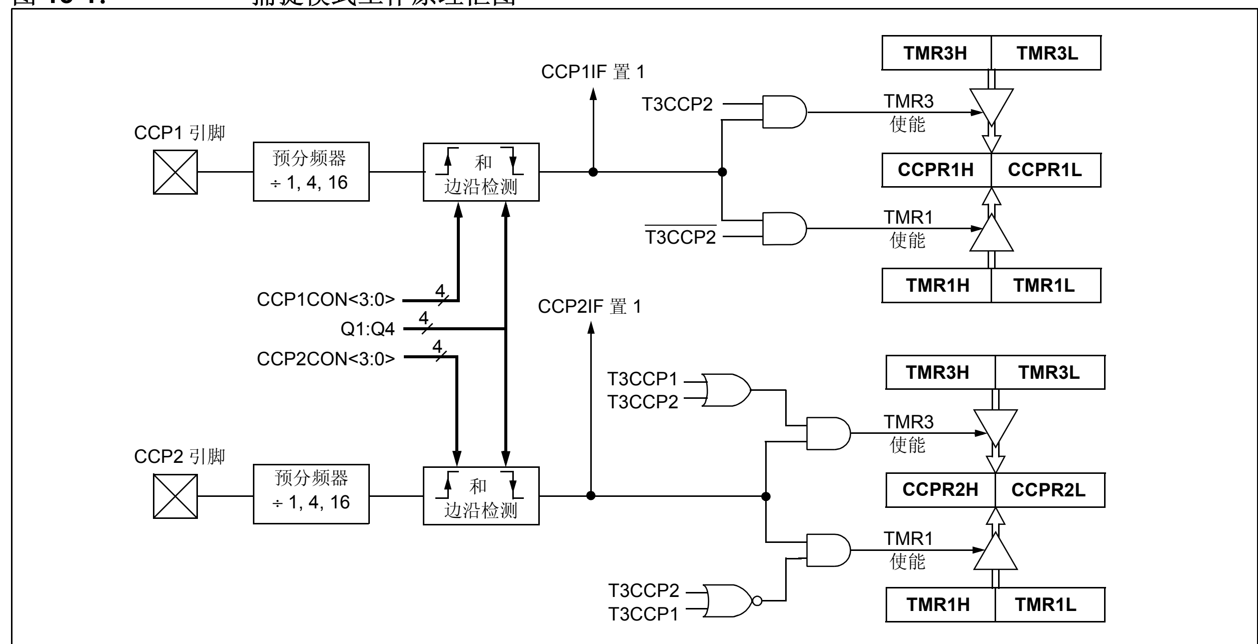
在捕捉模式下有 4 种预分频比设置，它们可作为工作模式的一部分由模式选择位（CCPxM3:CCPxM0）选择。每当关闭 CCP 模块或禁止捕捉模式时，预分频计数器就将被清零。这意味着任何复位都将清零预分频器计数器。

从一种捕捉预分频比切换到另一种捕捉预分频比可能会产生中断，且不会清零预分频计数器。切换后的第一次捕捉可能来自于一个非零的预分频器。例 15-1 是切换捕捉预分频比时建议采用的方法。该例也使预分频计数器清零且不会产生 “误”中断。

例 15-1： 改变捕捉预分频比值（以 **CCP2** 为例）

```
CLRF   CCP2CON      ; Turn CCP module off
MOVLW  NEW_CAPT_PS  ; Load WREG with the
                    ; new prescaler mode
                    ; value and CCP ON
MOVWF  CCP2CON      ; Load CCP2CON with
                    ; this value
```

图 15-1： 捕捉模式工作原理框图



15.3 比较模式

在比较模式下，16位CCPRx寄存器的值将不断与TMR1或 TMR3 寄存器对的值相比较。当两者匹配时， CCPx引脚可能会出现以下几种情况：

- 输出高电平
- 输出低电平
- 电平翻转 （高电平变为低电平或低电平变为高电平）
- 保持不变 （即反映 I/O 锁存器的状态）

引脚动作取决于模式选择位 （CCPxM3:CCPxM0）的值。同时，中断标志位 CCPxIF 置 1。

15.3.1 CCP 引脚配置

用户必须通过将相应的 TRIS 位清零，将 CCPx 引脚配置为输出。

注：	清零 CCP2CON 寄存器会将 RB3 或 RC1 比较输出锁存器 （取决于器件配置）强制为默认的低电平。这不是 PORTB 或 PORTC I/O 数据锁存器。

15.3.2 TIMER1/TIMER3 模式选择

如果 CCP 模块使用比较功能，Timer1 和 / 或 Timer3 必须运行在定时器模式或同步计数器模式下。在异步计数器模式下，无法进行比较操作。

15.3.3 软件中断模式

当选择了产生软件中断模式时 （CCPxM3:CCPxM0 = 1010），对应的 CCPx 引脚不受影响。只产生一个 CCP 中断 （如果中断被允许并把 CCPxIE 位置 1）。

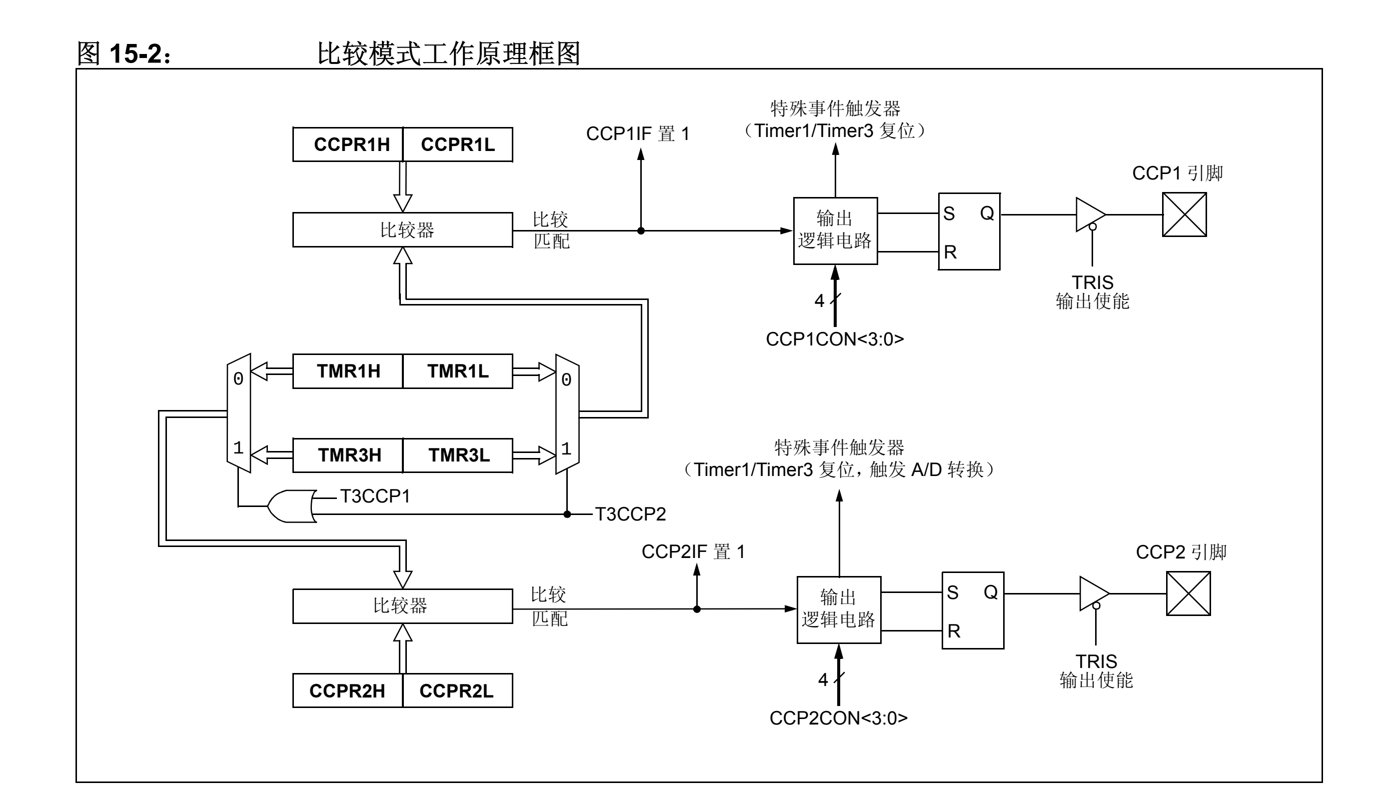
15.3.4 特殊事件触发器

两个 CCP 模块均配备了一个特殊事件触发器，在比较模式下可产生内部硬件信号以触发其他模块的动作。通过选择“比较特殊事件触发模式”（CCPxM3:CCPxM0 = 1011），使能特殊事件触发器。

对于任何一个 CCP 模块，无论当前使用哪个定时器作为模块的时基，特殊事件触发信号将把对应的定时寄存器对复位。这样 CCPRx 寄存器就可用作两个定时器的可编程周期寄存器。

CCP2 的特殊事件触发信号还能启动 A/D 转换。要实现此功能，必须首先使能 A/D 转换器。

图 15-2： 比较模式工作原理框图



 初稿

表 **15-3**： 与捕捉、比较、 **TIMER1** 和 **TIMER3** 相关的寄存器

名称	Bit 7	Bit 6	Bit 5	Bit 4	Bit 3	Bit 2	Bit 1	Bit 0	复位值所在页
INTCON	GIE/GIEH	PEIE/GIEL	TMR0IE	INT0IE	RBIE	TMR0IF	INT0IF	RBIF	51
RCON	IPEN	SBOREN(1)	—	$\overline{\text{RI}}$	$\overline{\text{TO}}$	$\overline{\text{PD}}$	$\overline{\text{POR}}$	$\overline{\text{BOR}}$	52
PIR1	SPPIF(2)	ADIF	RCIF	TXIF	SSPIF	CCP1IF	TMR2IF	TMR1IF	54
PIE1	SPPIE(2)	ADIE	RCIE	TXIE	SSPIE	CCP1IE	TMR2IE	TMR1IE	54
IPR1	SPPIP(2)	ADIP	RCIP	TXIP	SSPIP	CCP1IP	TMR2IP	TMR1IP	54
PIR2	OSCFIF	CMIF	USBIF	EEIF	BCLIF	HLVDIF	TMR3IF	CCP2IF	54
PIE2	OSCFIE	CMIE	USBIE	EEIE	BCLIE	HLVDIE	TMR3IE	CCP2IE	54
IPR2	OSCFIP	CMIP	USBIP	EEIP	BCLIP	HLVDIP	TMR3IP	CCP2IP	54
TRISB	TRISB7	TRISB6	TRISB5	TRISB4	TRISB3	TRISB2	TRISB1	TRISB0	54
TRISC	TRISC7	TRISC6	—	—	—	TRISC2	TRISC1	TRISC0	54
TMR1L	Timer1 寄存器的低字节								52
TMR1H	Timer1 寄存器的高字节								52
T1CON	RD16	T1RUN	T1CKPS1	T1CKPS0	T1OSCEN	$\overline{\text{T1SYNC}}$	TMR1CS	TMR1ON	52
TMR3H	Timer3 寄存器的高字节								53
TMR3L	Timer3 寄存器的低字节								53
T3CON	RD16	T3CCP2	T3CKPS1	T3CKPS0	T3CCP1	$\overline{\text{T3SYNC}}$	TMR3CS	TMR3ON	53
CCPR1L	捕捉 / 比较 /PWM 寄存器 1 的低字节								53
CCPR1H	捕捉 / 比较 /PWM 寄存器 1 的高字节								53
CCP1CON	P1M1(2)	P1M0(2)	DC1B1	DC1B0	CCP1M3	CCP1M2	CCP1M1	CCP1M0	53
CCPR2L	捕捉 / 比较 /PWM 寄存器 2 的低字节								53
CCPR2H	捕捉 / 比较 /PWM 寄存器 2 的高字节								53
CCP2CON	—	—	DC2B1	DC2B0	CCP2M3	CCP2M2	CCP2M1	CCP2M0	53

图注： — = 未用 （读为 `0`）。捕捉 / 比较、 Timer1 或 Timer3 不使用阴影单元。

注 **1**： 只有当 BOREN<1:0> = `01` 时， SBOREN 位才可用，否则该位读为 `0`。

2： 28 引脚器件不使用这些位，应始终保持这些位清零。

15.4 PWM 模式

在脉宽调制（Pulse-Width Modulation，PWM）模式下，CCPx 引脚可输出分辨率高达 10 位的 PWM 输出。由于 CCP2 引脚与 PORTB 或 PORTC 数据锁存器复用，因此必须清零相应的 TRIS 位才能使 CCP2 引脚成为输出引脚。

> **注：** 清零 CCP2CON 寄存器会将 RB3 或 RC1 输出锁存器（取决于器件配置）强制为默认的低电平状态。这不是 PORTB 或 PORTC I/O 数据锁存器。

图 15-3 所示为 CCP 模块在 PWM 模式时的简化框图。

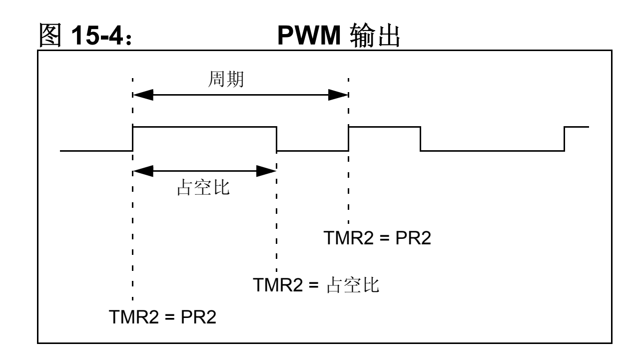
如需了解设置 CCP 模块以进行 PWM 操作的详细步骤，请参见第 **15.4.4 节 “设置 PWM 操作”**。

图 15-3： PWM 简化框图

注 1：8 位 TMR2 值与内部 Q 时钟或预分频器的 2 位一起构成 10 位时基。

PWM 输出（图 15-4）包含一个时基（周期）和一段输出高电平的时间（占空比）。PWM 信号的频率是周期的倒数（1/ 周期）。

图 15-4： PWM 输出



15.4.1 PWM 周期

PWM 周期可以通过写 PR2 寄存器来设定。使用以下公式计算 PWM 周期：

公式 15-1：

$$\text{PWM 周期} = [(PR2) + 1] \cdot 4 \cdot T_{OSC} \cdot (\text{TMR2 预分频值})$$

PWM 频率定义为 1 / [PWM 周期]。

当 TMR2 等于 PR2 时，在下一个计数周期中会发生以下三个事件：

- TMR2 被清零
- CCPx 引脚被置 1 （例外情况：如果 PWM 占空比 = 0%， CCPx 引脚将不被置 1）
- PWM 占空比从 CCPRxL 锁存到 CCPRxH

> **注：** PWM 频率不是由 Timer2 后分频器（见第 **13.0 节 “Timer2 模块”**）决定的。使用后分频器时，其伺服更新速率可与 PWM 输出频率不同。

15.4.2 PWM 占空比

PWM 占空比可通过写入 CCPRxL 寄存器和 CCPxCON<5:4> 位来设定。最高分辨率可达 10 位。CCPRxL 包含占空比的高 8 位，CCPxCON<5:4> 包含低 2 位。由 CCPRxL:CCPxCON<5:4> 表示这个 10 位值。计算占空比（以时间为单位）的公式如下：

公式 15-2：

$$\text{PWM 占空比} = (CCPRxL:CCPxCON<5:4>) \cdot T_{OSC} \cdot (\text{TMR2 预分频值})$$

可以在任何时候写入 CCPRxL 和 CCPxCON<5:4>，但直到 PR2 和 TMR2 发生匹配（即周期结束）时，占空比的值才被锁存到 CCPRxH 中。在 PWM 模式下，CCPRxH 是只读寄存器。

CCPRxH 寄存器和一个 2 位的内部锁存器用于为 PWM 占空比提供双重缓冲。这种双重缓冲结构非常重要，可以避免在 PWM 工作过程中产生毛刺。

当 CCPRxH 和 2 位锁存器的值与 TMR2 （连有内部 Q 时钟或 TMR2 预分频器的 2 位）匹配时，CCPx 引脚被清零。

对于给定的 PWM 频率，其最大分辨率（位）由右边的公式计算：

公式 **15-3：**

$$\text{PWM 分辨率（最大）} = \frac{\log\left(\frac{F_{OSC}}{F_{PWM}}\right)}{\log(2)} \text{位}$$

注：	如果 PWM 的占空比值大于 PWM 周期，CCPx 引脚将不会被清零。

表 15-4： 40 MHz 时的 PWM 频率与分辨率示例

PWM 频率	**2.44 kHz**	**9.77 kHz**	**39.06 kHz**	**156.25 kHz**	**312.50 kHz**	**416.67 kHz**
定时器预分频值（1、 4 或 16）	16	4	1	1	1	1
PR2 值	FFh	FFh	FFh	3Fh	1Fh	17h
最高分辨率（位）	10	10	10	8	7	6.58

15.4.3 PWM 自动关闭（仅 CCP1）

28 引脚器件的 CCP1 也具有增强型 CCP 模块的 PWM 自动关闭功能。在**第 16.4.7 节“增强型 PWM 自动关闭功能”**中对此功能的操作进行了详细说明。

CCP2 不可以使用自动关闭功能。

15.4.4 设置 PWM 操作

在为 CCP 模块配置 PWM 工作模式时应该遵循以下步骤：

1. 通过写 PR2 寄存器设置 PWM 周期。
2. 通过写 CCPRxL 寄存器和 CCPxCON<5:4> 位来设置 PWM 占空比。
3. 通过清零相应的 TRIS 位将 CCPx 引脚配置为输出。
4. 通过写 T2CON 来设置 TMR2 预分频值并使能 Timer2。
5. 配置 CCPx 模块使之工作于 PWM 模式。

表 **15-5:** 与 **PWM** 和 **TIMER2** 相关的寄存器

名称	Bit 7	Bit 6	Bit 5	Bit 4	Bit 3	Bit 2	Bit 1	Bit 0	复位值所在页
INTCON	GIE/GIEH	PEIE/GIEL	TMR0IE	INT0IE	RBIE	TMR0IF	INT0IF	RBIF	51
RCON	IPEN	SBOREN[1]	—	$\overline{RI}$	$\overline{TO}$	$\overline{PD}$	$\overline{POR}$	$\overline{BOR}$	52
PIR1	SPPIF[2]	ADIF	RCIF	TXIF	SSPIF	CCP1IF	TMR2IF	TMR1IF	54
PIE1	SPPIE[2]	ADIE	RCIE	TXIE	SSPIE	CCP1IE	TMR2IE	TMR1IE	54
IPR1	SPPIP[2]	ADIP	RCIP	TXIP	SSPIP	CCP1IP	TMR2IP	TMR1IP	54
TRISB	TRISB7	TRISB6	TRISB5	TRISB4	TRISB3	TRISB2	TRISB1	TRISB0	54
TRISC	TRISC7	TRISC6	—	—	—	TRISC2	TRISC1	TRISC0	54
TMR2	Timer2 寄存器								52
PR2	Timer2 周期寄存器								52
T2CON	—	T2OUTPS3	T2OUTPS2	T2OUTPS1	T2OUTPS0	TMR2ON	T2CKPS1	T2CKPS0	52
CCPR1L	捕捉 / 比较 /PWM 寄存器 1 的低字节								53
CCPR1H	捕捉 / 比较 /PWM 寄存器 1 的高字节								53
CCP1CON	P1M1[2]	P1M0[2]	DC1B1	DC1B0	CCP1M3	CCP1M2	CCP1M1	CCP1M0	53
CCPR2L	捕捉 / 比较 /PWM 寄存器 2 的低字节								53
CCPR2H	捕捉 / 比较 /PWM 寄存器 2 的高字节								53
CCP2CON	—	—	DC2B1	DC2B0	CCP2M3	CCP2M2	CCP2M1	CCP2M0	53
ECCP1AS	ECCPASE	ECCPAS2	ECCPAS1	ECCPAS0	PSSAC1	PSSAC0	PSSBD1[2]	PSSBD0[2]	53
ECCP1DEL	PRSEN	PDC6[2]	PDC5[2]	PDC4[2]	PDC3[2]	PDC2[2]	PDC1[2]	PDC0[2]	53

图注： — = 未用 （读为 0）。 PWM 或 Timer2 不使用阴影单元。

注 **1：** 只有当 BOREN<1:0> = 01 时， SBOREN 位才可用，否则该位读为 0。

2： 28 引脚器件不使用这些位，应始终保持这些位清零。

 初稿

16.0 增强型捕捉 / 比较 /PWM （ECCP）模块

注：	只有 40/44 引脚器件才具有 ECCP 模块。

在 PIC18F4455/4550 器件中， CCP1 模块为带有增强的 PWM 功能的标准 CCP 模块。这些增强的功能包括提供 2 路或 4 路输出通道、用户可选的极性、死区控制以及自动关闭和重启。第 **16.4 节“增强型 PWM 模式”**中有对增强功能的详细讨论。ECCP 模块的捕捉、比较和单输出 PWM 功能与标准 CCP 模块中描述的相同。

增强型 CCP 模块的控制寄存器如寄存器 16-1 所示。它与 PIC18F2255/2550 器件中的 CCPxCON 寄存器的不同之处在于它使用了两个最高有效位来控制 PWM 功能。

寄存器 16-1： CCP1CON：ECCP 控制寄存器 （40/44 引脚器件）

R/W-0	R/W-0	R/W-0	R/W-0	R/W-0	R/W-0	R/W-0	R/W-0
P1M1	P1M0	DC1B1	DC1B0	CCP1M3	CCP1M2	CCP1M1	CCP1M0
bit 7							bit 0

bit 7-6 **P1M1:P1M0**：增强型 PWM 输出配置位

如果 CCP1M3:CCP1M2 = 00、 01 或 10：
xx = P1A 被指定为捕捉 / 比较输入 / 输出； P1B、 P1C 和 P1D 被指定为端口引脚

如果 CCP1M3:CCP1M2 = 11：
00 = 单输出：P1A 为调制输出； P1B、 P1C 和 P1D 被指定为端口引脚
01 = 全桥正向输出：P1D 为调制输出； P1A 有效； P1B 和 P1C 无效
10 = 半桥输出：P1A 和 P1B 为带死区控制的调制输出； P1C 和 P1D 被指定为端口引脚
11 = 全桥反向输出：P1B 为调制输出； P1C 有效； P1A 和 P1D 无效

bit 5-4 **DC1B1:DC1B0**：PWM 占空比 bit 1 和 bit 0

捕捉模式：
未用。

比较模式：
未用。

PWM 模式：
这两位是 10 位 PWM 占空比的低 2 位。占空比的高 8 位在 CCPR1L 中。

bit 3-0 **CCP1M3:CCP1M0**：增强型 CCP 模式选择位

0000 = 捕捉 / 比较 /PWM 关闭 （复位 ECCP 模块）
0001 = 保留
0010 = 比较模式，匹配时输出电平翻转
0011 = 捕捉模式
0100 = 捕捉模式，在每个下降沿发生
0101 = 捕捉模式，在每个上升沿发生
0110 = 捕捉模式，每 4 个上升沿发生
0111 = 捕捉模式，每 16 个上升沿发生
1000 = 比较模式， CCP1 引脚初始化为低电平，比较匹配时输出高电平 （将 CCP1IF 置 1）
1001 = 比较模式， CCP1 引脚初始化为高电平，比较匹配时输出低电平 （将 CCP1IF 置 1）
1010 = 比较模式，仅产生软件中断， CCP1 引脚恢复到 I/O 状态
1011 = 比较模式，触发特殊事件 （CCP1 复位 TMR1 或 TMR3，将 CCP1IF 位置 1）
1100 = PWM 模式：P1A 和 P1C 为高电平有效； P1B 和 P1D 也为高电平有效
1101 = PWM 模式：P1A 和 P1C 为高电平有效； P1B 和 P1D 为低电平有效
1110 = PWM 模式：P1A 和 P1C 为低电平有效； P1B 和 P1D 为高电平有效
1111 = PWM 模式：P1A 和 P1C 为低电平有效； P1B 和 P1D 也为低电平有效

图注：			
R = 可读位	W = 可写位	U = 未用位，读为 0	
-n = 上电复位时的值	1 = 置 1	0 = 清零	x = 未知

除了通过 CCP1CON 寄存器获得扩展的模式范围之外，ECCP 模块还有 2 个与增强型 PWM 操作以及自动关闭功能相关的寄存器。它们是：

- ECCP1DEL （死区延时寄存器）
- ECCP1AS （自动关闭配置寄存器）

16.1 ECCP 输出和配置

每个增强型 CCP 模块至多有 4 路 PWM 输出，这取决于选定的操作模式。这些输出 （被称为 P1A 至 P1D）与 PORTC 和 PORTD 的 I/O 引脚复用。输出是否有效取决于选定的 CCP 操作模式。表 16-1 总结了引脚的分配情况。

要将 I/O 引脚配置为 PWM 输出，必须通过设置 P1M1:P1M0 以及 CCP1M3:CCP1M0 位来选择适当的 PWM 模式。还必须将端口引脚的相应的 TRISC 和 TRISD 方向位设置为输出。

16.1.1 ECCP 模块和定时器资源

与标准 CCP 模块相同， ECCP 模块可以使用 Timer1、Timer2 或 Timer3，具体选用哪个定时器取决于所选定的模式。Timer1 和 Timer3 可用于工作在捕捉或比较模式下的模块，而 Timer2 可用于工作在 PWM 模式下的模块。标准 CCP 模块和增强型 CCP 模块的互连方式与标准 CCP 模块的互连方式相同。**第 15.1.1 节“CCP 模块和定时器资源”**中提供了更多有关定时器资源的详细信息。

16.2 捕捉和比较模式

除了要在下面讨论的特殊事件触发器的操作，ECCP 模块的捕捉和比较模式操作与 CCP 相同。**第 15.2 节“捕捉模式”**和**第 15.3 节 “比较模式”**详细讨论了这些操作。

16.2.1 特殊事件触发器

ECCP 的特殊事件触发器输出会复位 TMR1 或 TMR3 寄存器对，具体复位哪对寄存器取决于当前选定的定时器资源。这将使 CCPR1H:CCPR1L 寄存器可以被用作 Timer1 或 Timer3 的 16 位可编程周期寄存器。

16.3 标准 PWM 模式

当被配置为单输出模式时， ECCP 模块的工作方式与**第 15.4 节“PWM 模式”**中描述的 PWM 模式下的标准 CCP 模块相同。这种模式有时也被称为“兼容的 CCP”模式，如表 16-1 所示。

> **注：** 当设置单输出 PWM 操作时，用户可以随意使用**第 15.4.4 节 “设置 PWM 操作”**或**第 16.4.9 节 “PWM 操作的设置”**中说明的步骤。后者更加常用但是只适用于单输出或多输出 PWM。

表 16-1： 各种 **ECCP1** 模式的引脚分配

ECCP 模式	CCP1CON 配置	RC2	RD5	RD6	RD7
所有 PIC18F4455/4550 器件：					
兼容的 CCP	00xx 11xx	CCP1	RD5/PSP5	RD6/PSP6	RD7/PSP7
双输出 PWM	10xx 11xx	P1A	P1B	RD6/PSP6	RD7/PSP7
四输出 PWM	x1xx 11xx	P1A	P1B	P1C	P1D

图注： x = 任意值。阴影单元表示在给定模式下 ECCP 不使用的引脚。

 初稿

16.4 增强型 PWM 模式

增强型 PWM 模式提供了更多的 PWM 输出选项以适应范围更广的控制应用。该模块是标准 CCP 模块的向下兼容版本，提供至多 4 路输出，即 P1A 到 P1D。用户还能够选择信号的极性（高电平有效或低电平有效）。通过分别设置 CCP1CON 寄存器的 P1M1:P1M0 和 CCP1M3:CCP1M0 位可配置模块的输出模式和极性。

图 16-1所示为PWM操作的简化框图。所有的控制寄存器都是双重缓冲的，并且在一个新的 PWM 周期开始时（Timer2 复位时的周期边界）被装载以防止在输出上出现毛刺。但 PWM 延时寄存器 ECCP1DEL 例外，该寄存器在占空比边界或周期边界（选择两者中首先出现的那个）被装载。由于采用缓冲技术，模块将不会立即启动，而是等待分配的定时器复位。这意味着增强型PWM 波形并不完全与标准的 PWM 波形吻合，而是偏移一个完整的指令周期 （4 TOSC）。

如前所述，用户必须为输出手动配置相应的 TRIS 位。

16.4.1 PWM 周期

PWM 周期可以通过写 PR2 寄存器来设定。可使用以下公式来计算 PWM 周期：

公式 16-1：

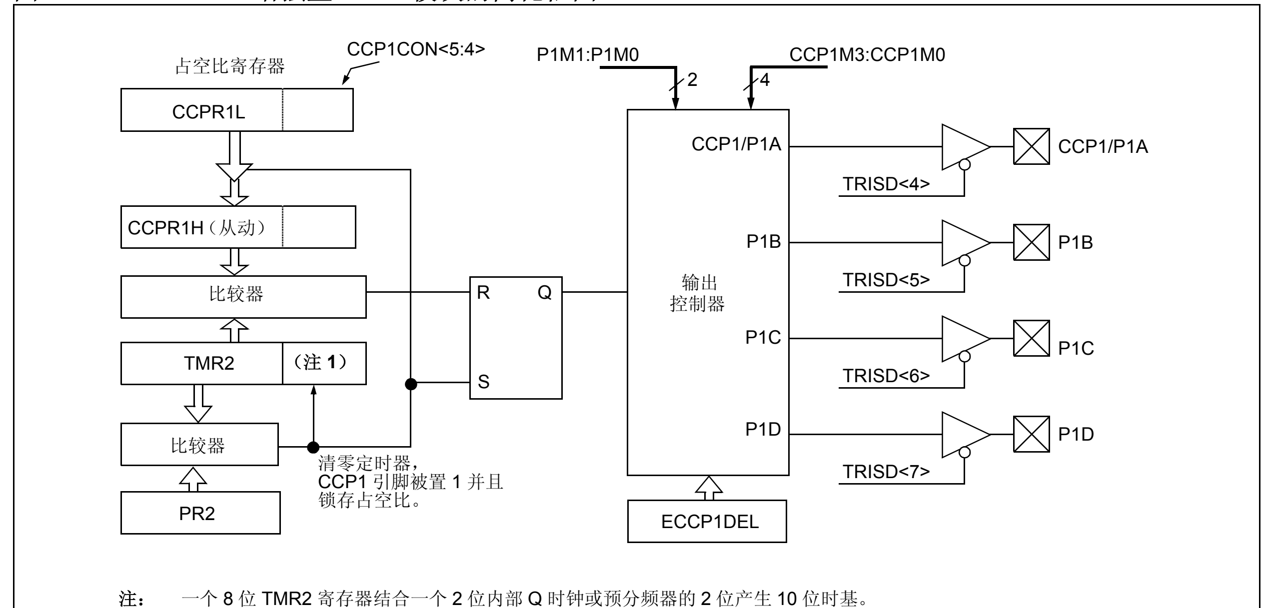
$$\text{PWM 周期} = [(PR2) + 1] \cdot 4 \cdot T_{OSC} \cdot (\text{TMR2 预分频值})$$

PWM 频率定义为 1 / [PWM 周期]。当 TMR2 等于 PR2 时，在下一个计数周期中会发生以下三个事件：

- TMR2 被清零
- CCP1 引脚被置 1 （如果 PWM 占空比 = 0%，CCP1 引脚将不会被置 1）
- PWM 占空比从 CCPR1L 复制到 CCPR1H

> **注：** PWM 频率不是由 Timer2 后分频器决定的（见第 **13.0 节“Timer2 模块”**）。使用后分频器时，其伺服更新速率可与 PWM 输出频率不同。

图 16-1： 增强型 PWM 模块的简化框图



注： 一个 8 位 TMR2 寄存器结合一个 2 位内部 Q 时钟或预分频器的 2 位产生 10 位时基。

 初稿

16.4.2 PWM 占空比

通过写 CCPR1L 寄存器和 CCP1CON<5:4> 位来设定 PWM 占空比。最高分辨率可达 10 位。CCPR1L 包含占空比的高 8 位，而 CCP1CON<5:4> 包含低 2 位。由 CCPR1L:CCP1CON<5:4> 表示完整的 10 位值。计算占空比的公式如下：

公式 16-2：

$$\text{PWM 占空比} = (\text{CCPR1L:CCP1CON<5:4>}) \cdot T_{OSC} \cdot (\text{TMR2 预分频值})$$

可以在任何时候写入 CCPR1L 和 CCP1CON<5:4>，但直到 PR2 和 TMR2 发生匹配 （即周期结束）时占空比值才会被锁存到 CCPR1H 中。在 PWM 模式中，CCPR1H 是只读寄存器。

CCPR1H 寄存器和一个 2 位的内部锁存器用于给 PWM 占空比提供双重缓冲。这种双重缓冲结构非常重要，可以避免在 PWM 工作过程中产生毛刺。当 CCPR1H 和 2 位锁存的值与 TMR2 （连有内部 2 位 Q 时钟或 TMR2 预分频器值中的 2 位）匹配时， CCP1 引脚被清零。对于给定的 PWM 频率，其最高 PWM 分辨率 （位）由右边的公式计算：

公式 16-3：

$$\text{PWM 分辨率（最大）} = \frac{\log\left(\frac{F_{OSC}}{F_{PWM}}\right)}{\log(2)} \text{ 位}$$

> **注：** 如果 PWM 占空比的值大于 PWM 周期，CCP1 引脚将不会被清零。

16.4.3 PWM 输出配置

CCP1CON 寄存器中的 P1M1:P1M0 位可以实现以下 4 种配置：

- 单输出
- 半桥输出
- 全桥输出，正向模式
- 全桥输出，反向模式

单输出模式是在**第 16.4 节 “增强型 PWM 模式”**中讨论的标准 PWM 模式。在接下来的各节中将详细介绍半桥和全桥输出模式。

图 16-2 和图 16-3 中汇总了各种配置下输出的大体关系。

表 16-2： 40 MHz 时的 PWM 频率和分辨率示例

PWM 频率	2.44 kHz	9.77 kHz	39.06 kHz	156.25 kHz	312.50 kHz	416.67 kHz
定时器预分频值 （1、 4 或 16）	16	4	1	1	1	1
PR2 值	FFh	FFh	FFh	3Fh	1Fh	17h
最高分辨率 （位）	10	10	10	8	7	6.58

图 16-2： PWM 输出关系 （高电平有效状态）

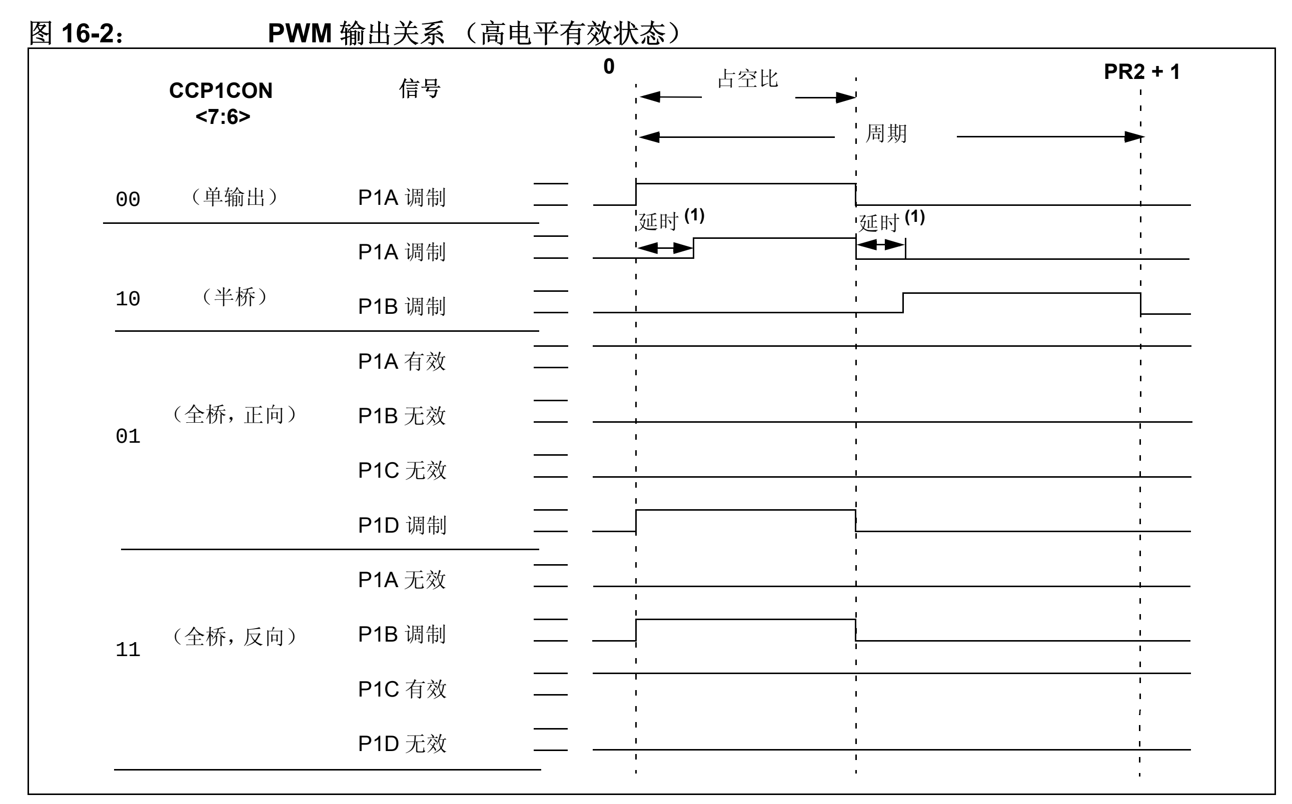
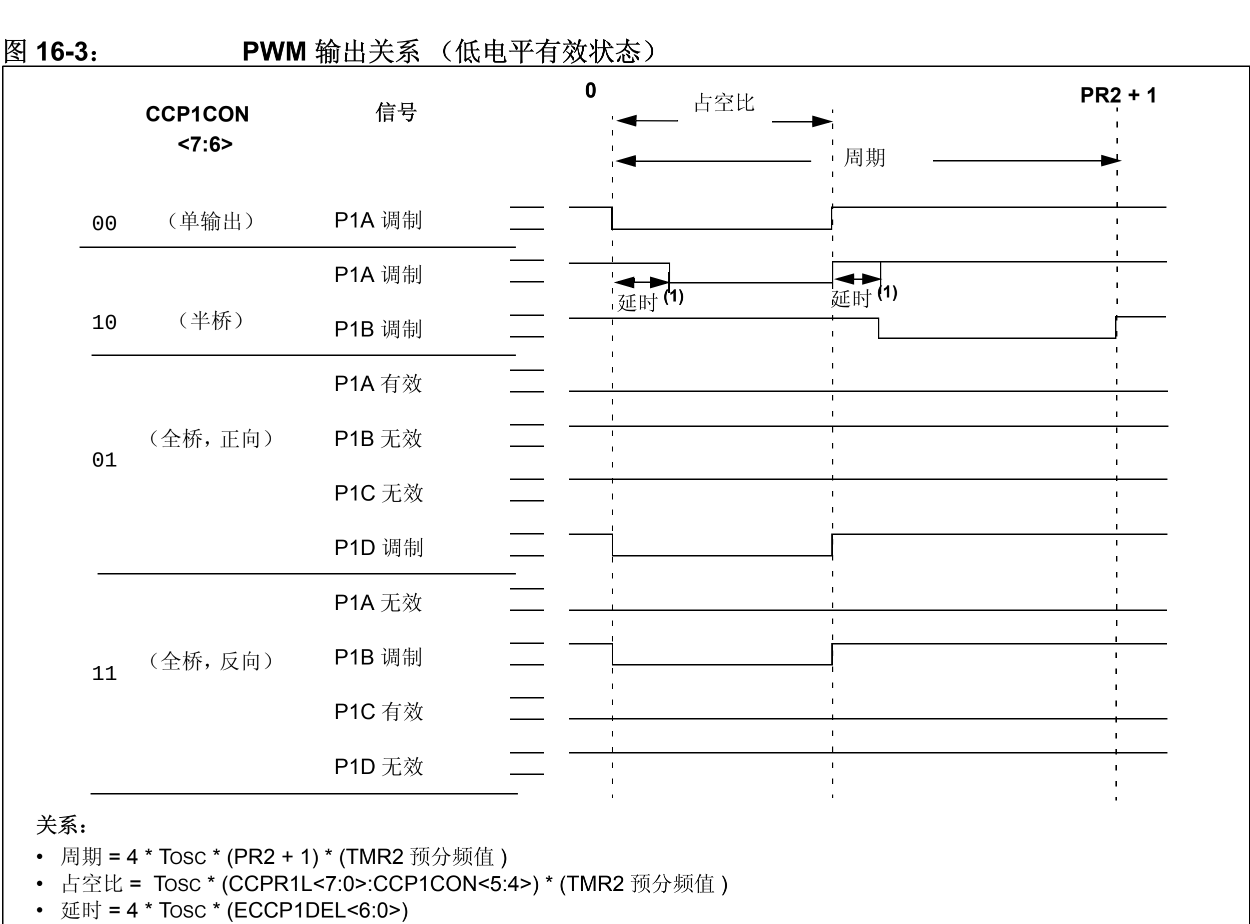


图 16-3： PWM 输出关系 （低电平有效状态）



关系：
- 周期 = 4 * Tosc * (PR2 + 1) * (TMR2 预分频值)
- 占空比 = Tosc * (CCPR1L<7:0>:CCP1CON<5:4>) * (TMR2 预分频值)
- 延时 = 4 * Tosc * (ECCP1DEL<6:0>)

注 1： 使用 ECCP1DEL 寄存器对死区延时进行编程设置 （**第 16.4.6 节 "可编程死区延迟"**）。

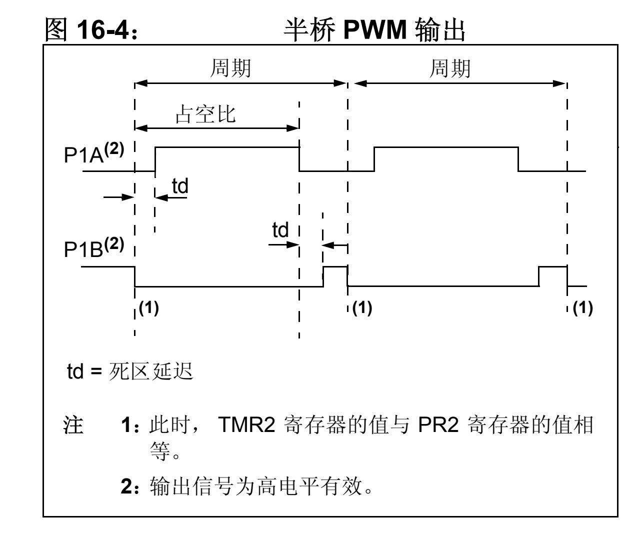
16.4.4 半桥模式

在半桥输出模式下，两个引脚用作输出端来驱动推挽式负载。PWM 输出信号从 P1A 引脚输出，而与之互补的 PWM 输出信号从 P1B 引脚输出（图 16-4）。如图 16-5 所示，此模式可用于半桥应用或那些使用 2 路 PWM 信号来调制 4 个功率开关的全桥应用。

在半桥输出模式下，可编程的死区延迟可用来防止在半桥功率器件中产生直通电流。PDC6:PDC0 位的值用来设置输出被驱动为有效之前的指令周期数。如果该值大于占空比，在整个周期内对应的输出将保持无效。欲知有关死区延迟操作的更多详细信息，请参见第 **16.4.6节“可编程死区延迟”**。

由于 P1A 和 P1B 输出与 PORTC<2> 和 PORTD<5> 数据锁存器复用，必须清零 TRISC<2> 和 TRISD<5> 位以将 P1A 和 P1B 配置为输出。

图 16-4： 半桥 **PWM** 输出



注 1：此时， TMR2 寄存器的值与 PR2 寄存器的值相等。
2：输出信号为高电平有效。

图 16-5： 半桥输出模式应用示例

标准半桥电路（“推挽式”）

半桥输出驱动全桥电路

16.4.5 全桥模式

在全桥输出模式下，4 个引脚被用作输出；但是，任一时刻只能有 2 个输出同时有效。在正向模式下，引脚 P1A 持续有效而引脚 P1D 为调制输出。在反向模式下，引脚 P1C 持续有效而引脚 P1B 为调制输出。这些在图 16-6 中进行了说明。

P1A、P1B、P1C 和 P1D 输出与 PORTC<2>、PORTD<5>、PORTD<6> 和 PORTD<7> 数据锁存器复用。必须清零 TRISC<2>、TRISD<5>、TRISD<6> 和 TRISD<7> 位以使 P1A、P1B、P1C 和 P1D 引脚作为输出引脚。

图 16-6： 全桥 **PWM** 输出

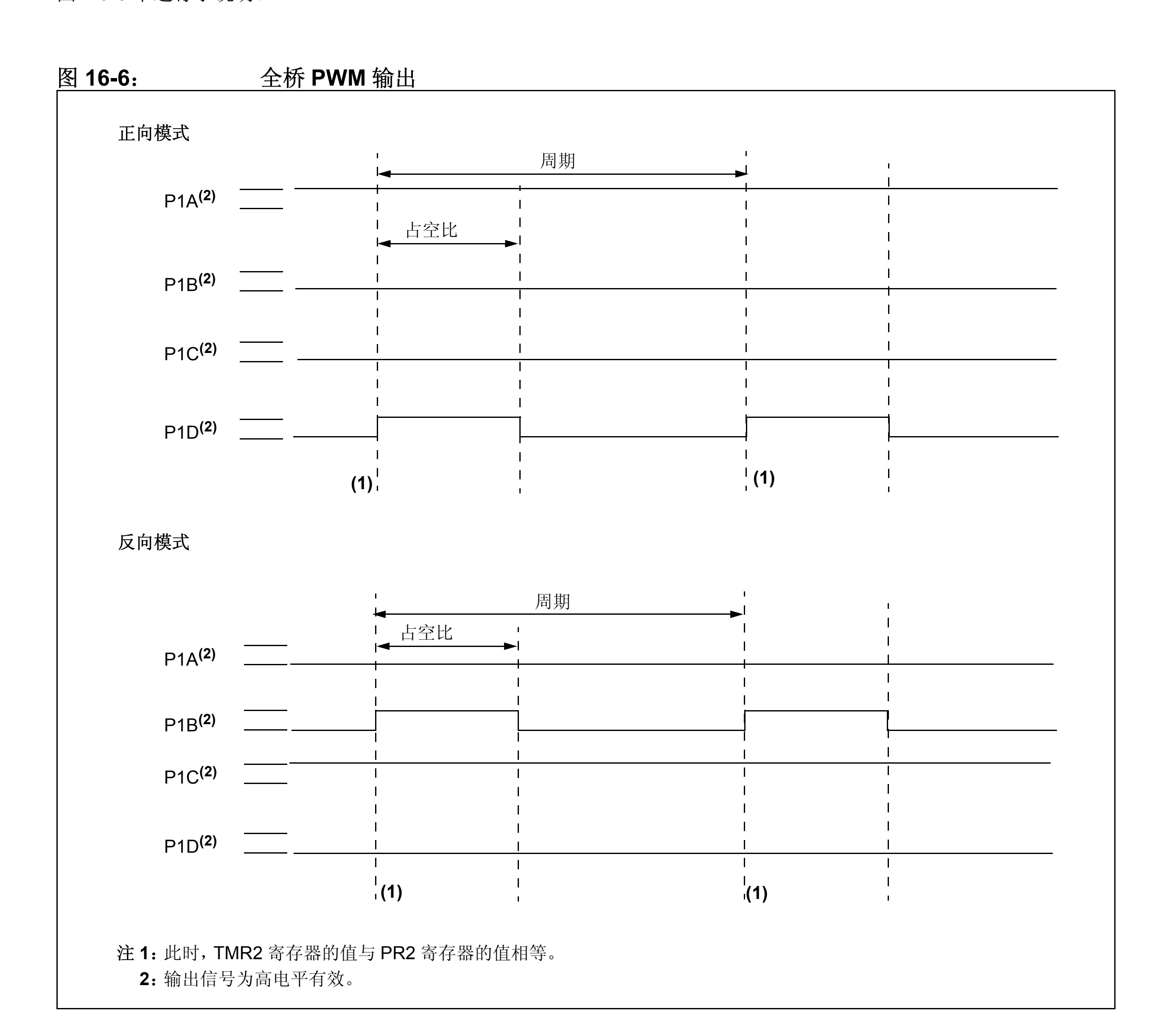
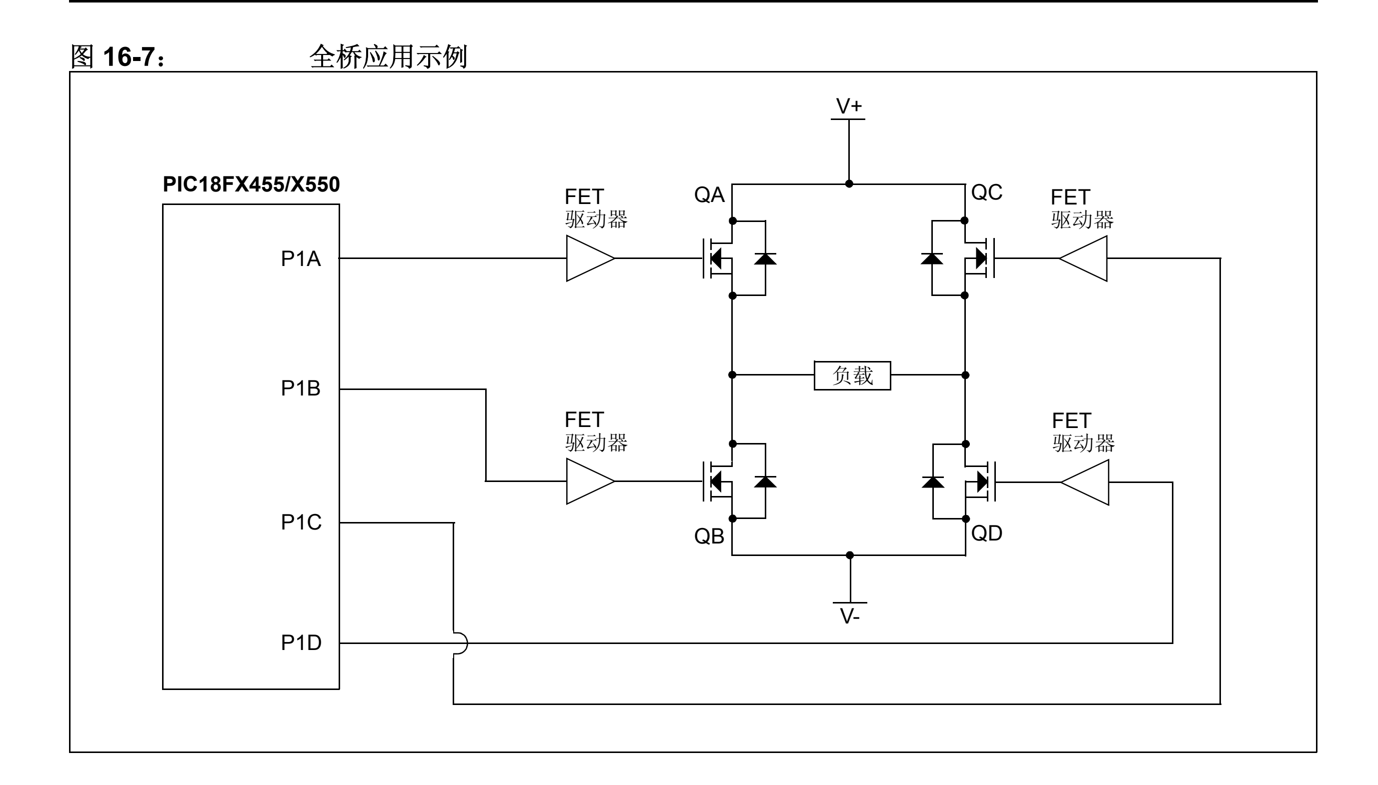


图 16-7： 全桥应用示例



16.4.5.1 全桥模式下的方向更改

在全桥输出模式下，CCP1CON 寄存器中的 P1M1 位使用户能控制负载中电流为正向还是反向。当应用固件更改此方向控制位时，模块将在下一个 PWM 周期采用新的方向。

就在当前PWM周期结束之前，调制输出（P1B和P1D）被置于它们的无效状态，而非调制输出（P1A 和 P1C）被切换到以相反的方向驱动负载。这发生在下一个 PWM 周期开始前的一段时间间隔内 (4 TOSC * (Timer2 预分频值))。Timer2 预分频器的分频值将是 1、4 或 16，这取决于 T2CKPS1:T2CKPS0 位 （T2CON<1:0>）的值。从切换非调制输出到下一个周期开始之间的这段时间内，调制输出 （P1B 和 P1D）保持无效。此关系如图 16-8 所示。

注意在全桥输出模式下，ECCP 模块不提供任何死区延迟。通常，由于在任何时间只调制一个输出，所以不需要死区延迟。然而，有一种情形可能需要死区延迟，即当以下两个条件同时满足的情况：

1. 当输出的占空比接近或等于 100% 时， PWM 输出的方向发生改变。
2. 功率开关（包括功率器件和驱动电路）的关断时间大于导通时间。

图 16-9 所示为在占空比接近 100% 时，PWM 方向从正向更改为反向的示例。在 t1 时刻，输出 P1A 和 P1D 变为无效，而输出 P1C 变为有效。在此示例中，由于功率器件的关断时间比导通时间长，直通电流可能在时间段 “t” 内流过功率器件 QC 和 QD （见图 16-7）。如果 PWM 方向从反向更改为正向，功率器件 QA 和 QB 将出现相同的现象。

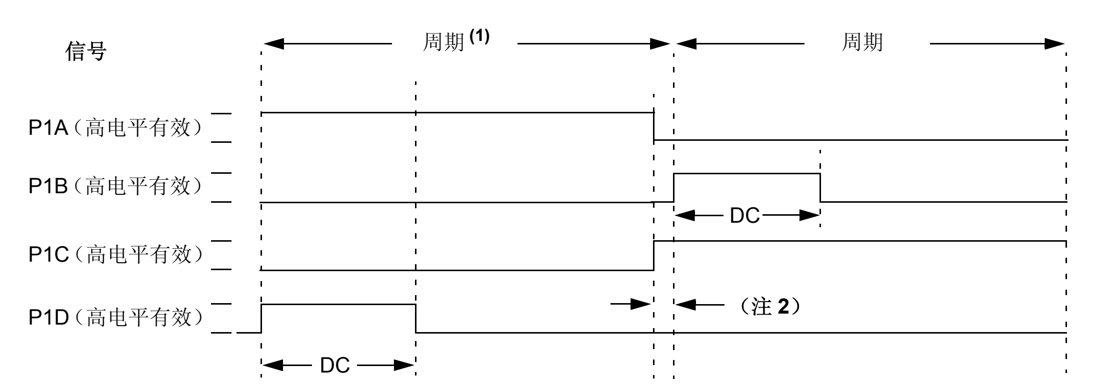
如果应用需要在高占空比情况下更改 PWM 方向，必须满足以下要求之一：

1. 在更改方向前的那个 PWM 周期缩小 PWM 的占空比。
2. 使用可使开关元件的关断速度比导通速度更快的开关驱动器。

可能还存在防止直通电流的其他方法。

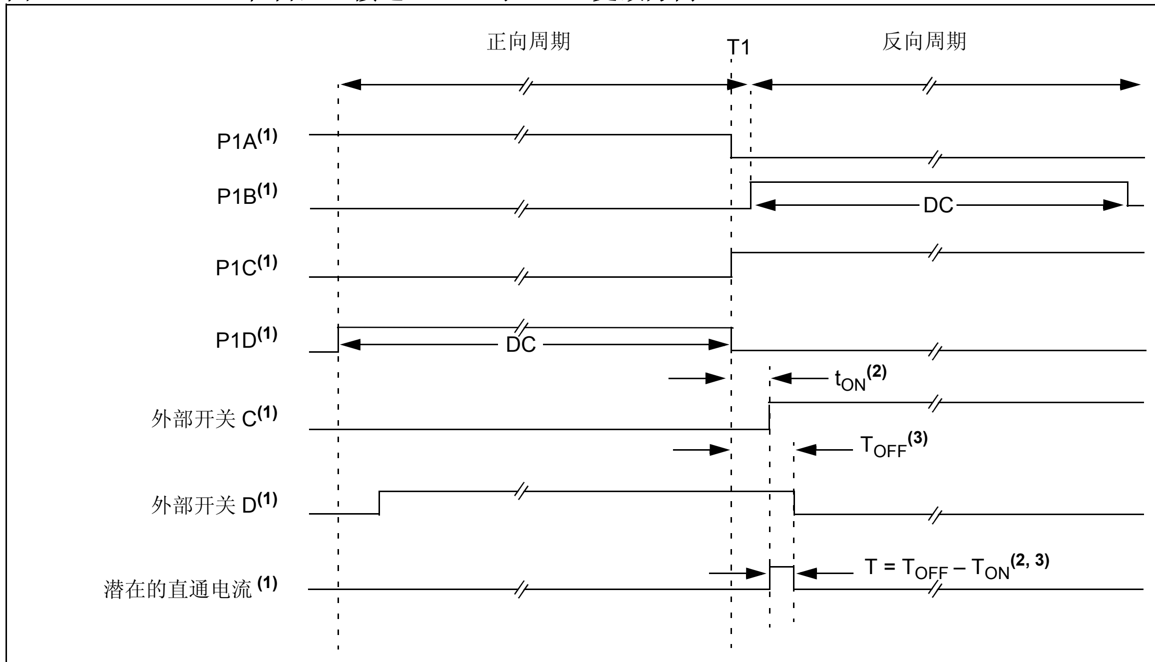


图 16-8： PWM 方向更改



注 1：可在 PWM 周期内的任何时间写 CCP1 控制寄存器中的方向位 （CCP1CON<7>）。
2：当更改方向时， P1A 和 P1C 上的信号在当前 PWM 周期结束之前的某个时间间隔内进行切换，该时间间隔可以是 4 TOSC、 16 TOSC 或 64 TOSC，这取决于 Timer2 的预分频值。在此期间被调制的 P1B 和 P1D 信号无效。

图 16-9： 在占空比接近 100% 时 PWM 更改方向



注 1：图示的所有信号为高电平有效。
2：t_{ON} 是功率开关 QC 及其驱动器的导通延迟时间。
3：t_{OFF} 是功率开关 QD 及其驱动器的关断延迟时间。

16.4.6 可编程死区延迟

注：	在带有标准 CCP 模块的 28 引脚器件中没有实现可编程死区延迟功能。

在半桥应用中，模块一直以 PWM 频率的调制信号驱动功率开关，关断功率开关通常比导通它需要更多的时间。如果上方和下方的功率开关同时切换 （一个打开，另一个关闭），两个开关可能会在一段很短时间内都处于打开状态，直到一个开关完全关断为止。在这很短的间隔内，很大的电流 （*直通电流*）可能流过两个功率开关，从而导致半桥供电电路短路。为了避免在切换期间流过这种潜在的破坏性直通电流，一般延迟导通其中的一个功率开关以使另一个开关完全关断。

在半桥输出模式下，可数字编程的死区延迟可用来避免直通电流破坏半桥的功率开关。该延迟在信号从非有效状态转换到有效状态时发生。参见图 16-4。ECCP1DEL 寄存器 （寄存器 16-2）的 PDC6:PDC0 位以单片机指令周期（TCY 或 4 TOSC）为单位设置延迟时间。由于标准 CCP 模块不支持半桥工作方式，因此在 28 引脚器件中不存在这些位。

16.4.7 增强型 PWM 自动关闭功能

当 ECCP 被编程设置为任何一种增强型 PWM 模式时，可以将有效输出引脚配置为自动关闭方式。当发生关闭事件时，增强型 PWM 输出引脚将被置于定义的关闭状态。

任一比较器模块、RB0/AN12/INT0/FLT0/SDI/SDA 引脚上的低电平或者以上三者的任意组合都可构成关闭事件。比较器可以用来监视与桥式电路中的电流成比例的电压输入。如果电压超过门限值，比较器将切换状态并触发关闭。另外， INT0 引脚上的数字信号也能触发关闭。通过不选择任何自动关闭源，可以禁止自动关闭功能。通过使用 ECCPAS2:ECCPAS0 位 （ECCP1AS<6:4>）选择将使用的自动关闭源。

当关闭事件发生时，输出引脚被陆续置于它们的关闭状态，关闭状态由 PSSAC1:PSSAC0 和 PSSBD1:PSSBD0 位（ECCP1AS3:ECCP1AS0）设定。每对引脚 （P1A/P1C 和 P1B/P1D）可被设置为驱动高电平、驱动低电平或三态（不驱动）。还需将 ECCPASE 位（ECCP1AS<7>）置 1 以便将增强型 PWM 输出保持在关闭状态。

当关闭事件发生时， ECCPASE 位被硬件置 1。如果不使能自动重启，当关闭源清除后， ECCPASE 位将被固件清零。如果使能自动重启，当自动关闭源被清除后，ECCPASE 位将被自动清零。

如果当 PWM 周期开始时 ECCPASE 位置 1， PWM 输出将在整个PWM周期内保持在关闭状态。当ECCPASE 位清零时， PWM 输出将在下一个 PWM 周期的开头恢复正常。

注：	当关闭条件有效时，禁止写ECCPASE位。

寄存器 16-2： ECCP1DEL：PWM 配置寄存器

R/W-0	R/W-0	R/W-0	R/W-0	R/W-0	R/W-0	R/W-0	R/W-0
PRSEN	PDC6(1)	PDC5(1)	PDC4(1)	PDC3(1)	PDC2(1)	PDC1(1)	PDC0(1)
bit 7							bit 0

bit 7 **PRSEN**：PWM 重启使能位
1 = 自动关闭时，一旦关闭事件被清除， ECCPASE 位立即自动清零； PWM 自动重启
0 = 自动关闭时，必须用软件清零 ECCPASE 位以重启 PWM

bit 6-0 **PDC6:PDC0**：PWM 延迟计数位 (1)
延迟时间为PWM信号变为有效的预计时间和实际时间之差，其单位为FOSC/4（4 * TOSC）周期。

注 1：在 28 引脚器件中，这些位是保留的，它们始终保持清零。

图注：			
R = 可读位	W = 可写位	U = 未用位，读为 0	
-n = 上电复位时的值	1 = 置 1	0 = 清零	x = 未知

寄存器 16-3： ECCP1AS：增强型捕捉 / 比较 /PWM 自动关闭控制寄存器

R/W-0	R/W-0	R/W-0	R/W-0	R/W-0	R/W-0	R/W-0	R/W-0
ECCPASE	ECCPAS2	ECCPAS1	ECCPAS0	PSSAC1	PSSAC0	PSSBD1[(1)]	PSSBD0[(1)]
bit 7							bit 0

bit 7 **ECCPASE**：ECCP 自动关闭事件状态位
1 = 已发生了关闭事件； ECCP 输出处于关闭状态
0 = ECCP 输出正常

bit 6-4 **ECCPAS2:ECCPAS0**：ECCP 自动关闭源选择位
111 = FLT0、比较器 1 或比较器 2
110 = FLT0 或比较器 2
101 = FLT0 或比较器 1
100 = FLT0
011 = 比较器 1 或 2
010 = 比较器 2 输出
001 = 比较器 1 输出
000 = 禁止自动关闭

bit 3-2 **PSSAC1:PSSAC0**：引脚 A 和 C 的关闭状态控制位
1x = 引脚 A 和 C 为三态 （40/44 引脚器件）
01 = 将引脚 A 和 C 驱动为 1
00 = 将引脚 A 和 C 驱动为 0

bit 1-0 **PSSBD1:PSSBD0**：引脚 B 和 D 的关闭状态控制位 [(1)]
1x = 引脚 B 和 D 为三态
01 = 将引脚 B 和 D 驱动为 1
00 = 将引脚 B 和 D 驱动为 0

注 **1**：在 28 引脚器件中，这些位是保留的，它们始终保持清零。

图注：			
R = 可读位	W = 可写位	U = 未用位，读为 0	
-n = 上电复位时的值	1 = 置 1	0 = 清零	x = 未知

16.4.7.1 自动关闭和自动重启

可以将自动关闭功能配置为在关闭事件后允许模块自动重启。通过置位 ECCP1DEL 寄存器的 PRSEN 位（ECCP1DEL<7>）使能此功能。

在 PRSEN = 1 （图 16-10）的关闭模式下，只要关闭源继续存在， ECCPASE 位将保持置 1。当关闭条件被清除时，ECCPASE 位清零。如果 PRSEN = 0（图 16-11），一旦出现关闭条件，ECCPASE 位将保持置 1 直到它被固件清零为止。一旦 ECCPASE 被清零，增强型 PWM 将在下一个 PWM 周期的开头恢复正常工作。

注：	当关闭条件有效时，禁止写ECCPASE位。

如果自动关闭源是某个比较器，则关闭条件为其输出的电平，而与 PRSEN 位的设置无关。只要关闭源继续存在， ECCPASE 位就不能被清零。

通过将 1 写入 ECCPASE 位可强制进入自动关闭模式。

16.4.8 启动注意事项

当在 PWM 模式下使用 ECCP 模块时，在硬件设计中必须在 PWM 输出引脚上使用适当的上拉和 / 或下拉电阻。当单片机退出复位状态时，所有的 I/O 引脚都处于高阻态。外部电路必须将功率开关保持在关闭状态，直到单片机用适当的信号电平驱动 I/O 引脚，或激活 PWM 输出为止。

CCP1M1:CCP1M0 位（CCP1CON<1:0>）允许用户为每对 PWM 输出引脚 （P1A/P1C 和 P1B/P1D）选择 PWM 输出信号是高电平有效还是低电平有效。必须在 PWM 引脚被配置为输出之前选择 PWM 输出的极性。建议不要在 PWM 引脚被配置为输出时更改极性配置，因为这可能造成应用电路的损坏。

当初始化 PWM 模块时，P1A、P1B、P1C 和 P1D 的输出锁存器可能处于不正确的状态。使能 ECCP 模块同时将 PWM 引脚使能为输出可能损坏应用电路。必须将 ECCP 模块使能为正确的输出模式并在经过一个完整的 PWM 周期之后，再将 PWM 引脚配置为输出。当第 2 个 PWM 周期开始时 TMR2IF 位会置 1，从而表明经过了一个完整的 PWM 周期。

图 16-10： PWM 自动关闭 （PRSEN = 1，使能自动重启）

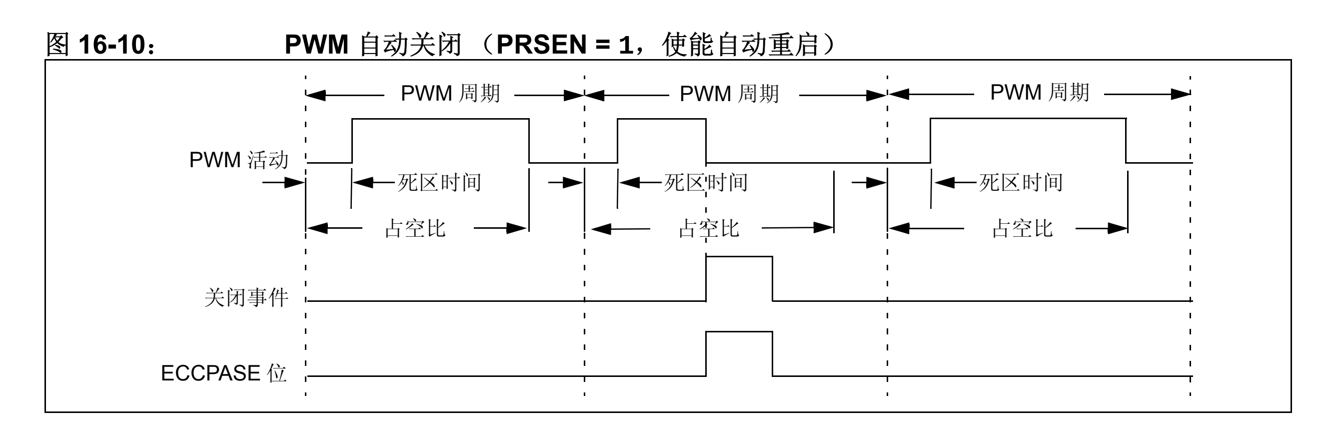
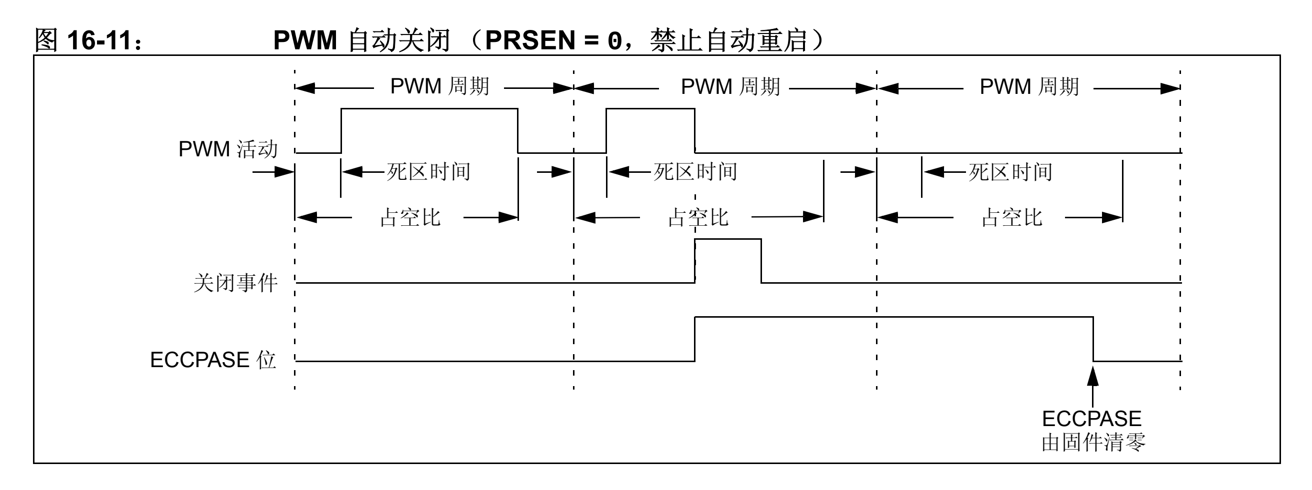


图 16-11： PWM 自动关闭 （PRSEN = 0，禁止自动重启）



 初稿

16.4.9 PWM 操作的设置

在将 ECCP 模块配置为 PWM 操作模式时应该执行以下步骤：

1. 通过将相应的 TRIS 位置 1，将 PWM 引脚 P1A 和 P1B （若使用 P1C 和 P1D，还包括这两个引脚）配置为输入。
2. 通过装载 PR2 寄存器设置 PWM 周期。
3. 如果需要设置自动关闭功能，请进行以下操作：
   - 禁止自动关闭 （ECCP1AS = 0）
   - 配置自动关闭源 （FLT0、比较器 1 或比较器 2）
   - 等待不关闭条件
4. 将 ECCP 模块配置为所需的 PWM 模式，通过用相应的值装载 CCP1CON 寄存器来完成该配置：
   - 通过 P1M1:P1M0 位选择输出配置和方向。
   - 通过 CCP1M3:CCP1M0 位选择 PWM 输出信号的极性。
5. 可通过装载 CCPR1L 寄存器和 CCP1CON<5:4> 位来设置 PWM 占空比。
6. 对于半桥输出模式，通过用合适的值装载 ECCP1DEL<6:0> 来设置死区延时。
7. 如果需要自动关闭操作，则装载 ECCP1AS 寄存器：
   - 使用 ECCPAS2:ECCPAS0 位选择自动关闭源。
   - 使用 PSSAC1:PSSAC0 和 PSSBD1:PSSBD0 位选择 PWM 输出引脚的关闭状态。
   - 将 ECCPASE 位 （ECCP1AS<7>）置 1。
   - 使用 CMCON 寄存器配置比较器。
   - 将比较器的输入端配置为模拟输入。
8. 如果需要自动重启操作，则将 PRSEN 位 （ECCP1DEL<7>）置 1。
9. 配置并启动 TMR2：
   - 清零 TMR2IF 中断标志位 （PIR1<1>）。
   - 通过装载 T2CKPS 位 （T2CON<1:0>）来设置 TMR2 预分频值。
   - 通过将 TMR2ON 位 （T2CON<2>）置 1 来使能 Timer2。
10. 在新的 PWM 周期开始后，使能 PWM 输出：
    - 等待 TMRn 溢出 （TMRnIF 位置 1）。
    - 通过分别清零相应的 TRIS 位，将 CCP1/P1A、P1B、 P1C 和 / 或 P1D 引脚使能为输出。
    - 将 ECCPASE 位 （ECCP1AS<7>）清零。

16.4.10 在功耗管理模式下的工作方式

在休眠模式下，禁止所有时钟源。Timer2 将不会递增并且模块的状态将不发生变化。如果 ECCP 引脚有输出，在休眠模式下将继续保持该输出值不变。当器件被唤醒时，它将开始继续工作。如果使能了双速启动，INTOSC 和后分频器的初始起振频率可能不会太稳定。

在 PRI_IDLE 模式下，主时钟将继续为 ECCP 模块提供时钟。在其他功耗管理模式下，选定的功耗管理模式时钟将为 Timer2 提供时钟。功耗管理模式下所使用的时钟的频率很可能与主时钟频率不同。

16.4.10.1 使用故障保护时钟监视器时的工作方式

如果使能了故障保护时钟监视器，时钟故障将强制器件进入 RC_RUN 功耗管理模式并且 OSCFIF（PIR2<7>）位将置 1。然后将由内部振荡器时钟源为 ECCP 提供时钟，该时钟的频率与主时钟不同。

如需更多详细信息，请参见前面的章节。

16.4.11 复位的影响

上电复位和后续的复位会将所有端口强制为输入模式，并强制 CCP 寄存器进入复位状态。

这会将增强型 CCP 模块强制性复位为与标准 CCP 模块兼容的状态。

表 16-3： 与 ECCP 模块和 TIMER1 到 TIMER3 相关的寄存器

名称	Bit 7	Bit 6	Bit 5	Bit 4	Bit 3	Bit 2	Bit 1	Bit 0	复位值所在页
INTCON	GIE/GIEH	PEIE/GIEL	TMR0IE	INT0IE	RBIE	TMR0IF	INT0IF	RBIF	51
RCON	IPEN	SBOREN[1]	—	$\overline{RI}$	$\overline{TO}$	$\overline{PD}$	$\overline{POR}$	$\overline{BOR}$	52
IPR1	SPPIP[2]	ADIP	RCIP	TXIP	SSPIP	CCP1IP	TMR2IP	TMR1IP	54
PIR1	SPPIF[2]	ADIF	RCIF	TXIF	SSPIF	CCP1IF	TMR2IF	TMR1IF	54
PIE1	SPPIE[2]	ADIE	RCIE	TXIE	SSPIE	CCP1IE	TMR2IE	TMR1IE	54
IPR2	OSCFIP	CMIP	USBIP	EEIP	BCLIP	HLVDIP	TMR3IP	CCP2IP	54
PIR2	OSCFIF	CMIF	USBIF	EEIF	BCLIF	HLVDIF	TMR3IF	CCP2IF	54
PIE2	OSCFIE	CMIE	USBIE	EEIE	BCLIE	HLVDIE	TMR3IE	CCP2IE	54
TRISB	TRISB7	TRISB6	TRISB5	TRISB4	TRISB3	TRISB2	TRISB1	TRISB0	54
TRISC	TRISC7	TRISC6	—	—	—	TRISC2	TRISC1	TRISC0	54
TRISD[2]	TRISD7	TRISD6	TRISD5	TRISD4	TRISD3	TRISD2	TRISD1	TRISD0	54
TMR1L	Timer1 寄存器的低字节								52
TMR1H	Timer1 寄存器的高字节								52
T1CON	RD16	T1RUN	T1CKPS1	T1CKPS0	T1OSCEN	$\overline{T1SYNC}$	TMR1CS	TMR1ON	52
TMR2	Timer2 模块寄存器								52
T2CON	—	T2OUTPS3	T2OUTPS2	T2OUTPS1	T2OUTPS0	TMR2ON	T2CKPS1	T2CKPS0	52
PR2	Timer2 周期寄存器								52
TMR3L	Timer3 寄存器的低字节								53
TMR3H	Timer3 寄存器的高字节								53
T3CON	RD16	T3CCP2	T3CKPS1	T3CKPS0	T3CCP1	$\overline{T3SYNC}$	TMR3CS	TMR3ON	53
CCPR1L	捕捉 / 比较 /PWM 寄存器 1 的低字节								53
CCPR1H	捕捉 / 比较 /PWM 寄存器 1 的高字节								53
CCP1CON	P1M1[2]	P1M0[2]	DC1B1	DC1B0	CCP1M3	CCP1M2	CCP1M1	CCP1M0	53
ECCP1AS	ECCPASE	ECCPAS2	ECCPAS1	ECCPAS0	PSSAC1	PSSAC0	PSSBD1[2]	PSSBD0[2]	53
ECCP1DEL	PRSEN	PDC6[2]	PDC5[2]	PDC4[2]	PDC3[2]	PDC2[2]	PDC1[2]	PDC0[2]	53

图注： — = 未用 （读为 0）。 ECCP 操作过程中不使用阴影单元。

注 1： 只有当 BOREN<1:0> = 01 时， SBOREN 位才可用；否则，此位读为 0。

2： 28 引脚器件不使用这些位或寄存器，应始终保持它们清零。

17.0 通用串行总线 （USB）

本节将详细讨论 USB 外设。由于该模块的特性与其他串口不同，所以需要读者对USB有所了解。**第 17.10节 “USB 概述”**提供了一些有关 USB 的高级信息，仅供设计参考。如果设计人员需要了解相关的最新信息，建议参考由 USB 设计论坛（USB Implementers Forum，USB-IF）出版的官方规范。USB 规范修订版 2.0 是此文档出版时的最新规范。

17.1 USB 外设概述

PIC18FX455/X550 系列器件包含全速和低速兼容的 USB 串行接口引擎 （Serial Interface Engine， SIE），它允许在任何 USB 主机和 PIC® 单片机之间进行高速通信。可以利用内部收发器或通过外部收发器将 SIE 直接连接到 USB。内部的 3.3V 稳压器也可以在 5V 的应用中被用作内部收发器的电源。

USB 模块还包含了某些特殊硬件以提高其工作性能。在器件的数据存储空间 （USB RAM）中提供了双端口存储器，以便单片机和 SIE 之间可以直接进行存储器访问。此外，还提供了缓冲器描述符，允许用户任意设置 USB RAM 空间中供端点使用的存储区。此外还提供了并行通信端口以方便大量数据的不间断传输，比如将数据等时传送到外部存储器缓冲器。

图 17-1 给出了 USB 外设及其功能的概况。

图 17-1： USB 外设和选项



注 1： 此信号仅在禁止内部收发器 （UTRDIS = 1）时可用。
2： 如果使用了外部上拉电阻，就应该禁止内部上拉电阻 （UPUEN = 0）。
3： 在使用外部 3.3V 电源时不要使能内部稳压器。

17.2 USB 状态和控制

USB 模块的工作状态是通过三个控制寄存器配置和管理的。除此之外，实际上共使用了19个寄存器管理USB通信。这些寄存器为：

- USB 控制寄存器 （UCON）
- USB 配置寄存器 （UCFG）
- USB 传输状态寄存器 （USTAT）
- USB 设备地址寄存器 （UADDR）
- 帧编号寄存器 （UFRMH:UFRML）
- 端点使能寄存器 0 到 15 （UEPn）

17.2.1 USB 控制寄存器 （UCON）

USB 控制寄存器 （寄存器 17-1）用于控制模块在传输过程中的操作。此寄存器包含控制以下行为的配置位：

- 使能主 USB 外设
- 乒乓 （Ping-Pang）缓冲器指针复位
- 挂起模式控制
- 禁止包传输

此外，USB 控制寄存器还包含一个状态位 SE0（UCON<5>），它用于表示总线上是否出现了单端 0。当使能了 USB 模块时，应该监视此位以确定差分数据线是否出现了单端 0 条件。这有助于区别初始上电状态和 USB 复位信号。

USB 模块的整体操作由 USBEN 位 （UCON<3>）控制。将此位置 1 可以激活模块并将缓冲描述符表中的所有 PPBI 位复位为 0。此位也可以激活片上稳压器并启用内部上拉电阻 （如果使能）。因此，此位可实现用软件控制与 USB 的连接或断开。虽然当此位被清零时，所有的状态和配置位都可被忽略，但将此位置 1 前还是需要预先对模块进行全面的配置。

寄存器 17-1： UCON：USB 控制寄存器

U-0	R/W-0	R-x	R/C-0	R/W-0	R/W-0	R/W-0	U-0
—	PPBRST	SE0	PKTDIS	USBEN	RESUME	SUSPND	—
bit 7							bit 0

bit 7 未用：读为 0

bit 6 **PPBRST：**乒乓缓冲器复位位
1 = 将所有乒乓缓冲器指针复位到缓冲描述符 （Buffer Descriptor， BD）的偶数存储区
0 = 不复位乒乓缓冲器指针

bit 5 **SE0：**有效单端 0 标志位
1 = 在 USB 总线上出现单端 0
0 = 未检测到单端 0

bit 4 **PKTDIS：**包传输禁止位
1 = 禁止 SIE 令牌和包处理，当接收到 SETUP 令牌时自动置 1
0 = 使能 SIE 令牌和包处理

bit 3 **USBEN：**USB 模块使能位
1 = 使能 USB 模块和支持电路 （与设备连接）
0 = 禁止 USB 模块和支持电路 （与设备断开）

bit 2 **RESUME：**恢复信令使能位
1 = 使能恢复信令
0 = 禁止恢复信令

bit 1 **SUSPND：**挂起 USB 位
1 = USB 模块和支持电路处于低功耗模式， SIE 时钟无效。
0 = USB 模块和支持电路处于正常工作模式， SIE 时钟按照所配置的频率提供时钟信号。

bit 0 未用：读为 0

图注：	R = 可读位		W = 可写位
C = 可清零位	U = 未用位，读为 0		
-n = 上电复位时的值	1 = 置 1	0 = 清零	x = 未知

PPBRST 位（UCON<6>）用于控制使用乒乓缓冲模式时的复位状态。当 PPBRST 位被置 1 时，所有乒乓缓冲器指针都被设置为指向偶数缓冲器。必须用固件清零 PPBRST 位。在不使用乒乓缓冲的模式中，此位被忽略。

PKTDIS 位（UCON<4>）是表示 SIE 禁止包传输和接收的标志位。当接收到 SETUP 令牌时，此位被 SIE 置 1，以允许处理 SETUP。单片机不能将此位置 1，只能将此位清零，以允许 SIE 继续发送和 / 或接收。缓冲描述符表中的任何待处理事件都仍然可用，这些事件由 USTAT 寄存器的 FIFO 缓冲器指示。

RESUME 位（UCON<2>）允许外设通过执行恢复信令来进行远程唤醒。要进行有效的远程唤醒，固件必须将 RESUME 位置 1 并保持此状态 10 ms，然后将此位清零。欲知更多有关恢复信令的信息，请参见 USB 2.0 规范中的第 7.1.7.5 节、第 11.9 节和第 11.4.4 节。

SUSPND 位（UCON<1>）将模块和支持电路（即稳压器）置于低功耗模式。SIE 的输入时钟也被禁止。响应 IDLEIF 中断时，该位必须由软件置位。在产生 ACTVIF 中断之后，由固件将此位复位。当此位有效时，设备与与总线处于连接状态，但是收发器输出处于空闲状态。V_{USB} 引脚上的电压根据此位值的不同而变化。在 IDLEIF 请求之前将此位置 1，将导致不可预料的总线行为。

> **注：** 处于挂起模式时，典型总线供电的 USB 设备的最大电流被限制为 500 µA。该电流是 PICmicro 器件及其支持电路总的汲取电流。在设备进入挂起模式时应该特别小心以确保汲取电流最小。

17.2.2 USB 配置寄存器（UCFG）

在通过 USB 通信之前，必须对与模块相关的内部和 / 或外部硬件进行配置。大部分配置是使用 UCFG 寄存器（寄存器 17-2）进行的。独立的 USB 稳压器（见第 **17.2.2.8 节“内部稳压器”**）也是通过此配置寄存器控制的。

UFCG 寄存器中的控制位可用于控制 USB 模块的绝大部分系统级行为，包括：

- 总线速度（全速和低速）
- 片上上拉电阻使能
- 片上收发器使能
- 乒乓缓冲器的使用

UCFG 寄存器还包含两个位，用于辅助模块测试、调试以及 USB 认证。这些位控制输出使能状态监视和眼图（Eye Pattern）产生。

> **注：** 只能在模块设置阶段配置 USB 速度、收发器和上拉电路。建议不要在使能模块时更改这些设置。

17.2.2.1 内部收发器

USB 外设有一个内置的 USB 2.0 全速和低速兼容的收发器，在 USB 内部与 SIE 相连。此功能部件在低成本的单芯片应用中很有用。UTRDIS 位（UCFG<3>）控制收发器，它在默认情况下是使能的（UTRDIS = 0）。FSEN 位（UCFG<2>）控制收发器的速度，将此位置 1 将使能全速工作模式。

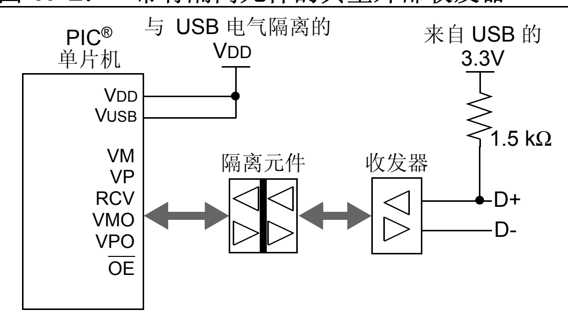
片上 USB 上拉电阻由 UPUEN 位（UCFG<4>）控制。只有在使能了片上收发器的时候才能选择这些上拉电阻。

USB 规范要求使用 3.3V 的电压进行通信，但是，USB 模块的其他部分需要在更高的电压下运行。因此，由独立的电源 V_{USB} 为收发器供电。

17.2.2.2 外部收发器

此模块为片外收发器提供支持。在因为物理条件限制而要求收发器远离 SIE 的应用中要用到片外收发器。例如，需要与 USB 电气隔离的应用可以通过某些隔离元件将外部收发器连接到单片机的 SIE（图 17-2）。通过将 UTRDIS 位置 1 可以使能外部收发器。

图 17-2： 带有隔离元件的典型外部收发器



注： 上图为全速配置的简单示意图，使用的是带有隔离元件的外部收发器。

寄存器 17-2： UCFG：USB 配置寄存器

R/W-0	R/W-0	U-0	R/W-0	R/W-0	R/W-0	R/W-0	R/W-0
UTEYE	UOEMON(1)	—	UPUEN(2,3)	UTRDIS(2)	FSEN(2)	PPB1	PPB0
bit 7							bit 0

bit 7 **UTEYE**：USB 眼图测试使能位
1 = 使能眼图测试
0 = 禁止眼图测试

bit 6 **UOEMON**：USB $\overline{OE}$ 监视器使能位 (1)
1 = $\overline{OE}$ 信号有效；它表示驱动 D+/D- 线的时间间隔
0 = $\overline{OE}$ 信号无效

bit 5 未用：读为 0

bit 4 **UPUEN**：USB 片上上拉使能位 (2,3)
1 = 使能片上上拉电路 （当 FSEN = 1 时， D+ 上拉或当 FSEN = 0 时， D- 上拉）
0 = 禁止片上上拉电路

bit 3 **UTRDIS**：片上收发器禁止位 (2)
1 = 禁止片上收发器；使能数字收发器接口
0 = 使能片上收发器

bit 2 **FSEN**：全速使能位 (2)
1 = 全速器件：控制收发器边沿速率，要求输入时钟为 48 MHz
0 = 低速器件：控制收发器边沿速率，要求输入时钟为 6 MHz

bit 1-0 **PPB1:PPB0**：乒乓缓冲器配置位
11 = 保留
10 = 使能所有端点的奇偶乒乓缓冲器
01 = 使能输出端点 0 的奇偶乒乓缓冲器
00 = 禁止奇 / 偶乒乓缓冲器

注 **1**：如果 UTRDIS 置 1，无论 UOEMON 位的设置如何， $\overline{OE}$ 信号都将有效。
2：UPUEN、UTRDIS 和 FSEN 位在 USB 模块使能期间不能更改。必须在使能模块之前预先配置这些值。
3：仅当使能片上收发器时 （UTRDIS = 0）此位有效；否则，可被忽略。

图注：			
R = 可读位	W = 可写位	U = 未用位，读为 0	
-n = 上电复位时的值	1 = 置 1	0 = 清零	x = 未知

模块中有 6 条信号线用来与外部收发器进行通信并控制外部收发器：

- VM：单端 D- 线路输入
- VP：单端 D+ 线路输入
- RCV：差分接收器输入
- VMO：差分线路驱动器输出
- VPO：差分线路驱动器输出
- $\overline{OE}$：输出使能

VPO 和 VMO 信号是由 SIE 输出到外部收发器的。RCV 信号是由外部收发器输出到 SIE 的，它表示将串行总线上的差分信号转换为一串单脉冲序列。VM 和 VP 信号将串行总线上不能用 RCV 信号捕捉的情况报告给 SIE。这些信号的组合状态和它们的说明在表 17-1和表 17-2 中列出。

表 17-1： 到收发器的差分输出

VPO	VMO	总线状态
0	0	单端 0
0	1	差分 0
1	0	差分 1
1	1	非法条件

表 17-2： 来自收发器的单端输入

VP	VM	总线状态
0	0	单端 0
0	1	低速
1	0	高速
1	1	错误

$\overline{OE}$ 信号触发外部收发器的状态。器件将此线拉为低电平，将使能从 SIE 到外部器件的数据传输。

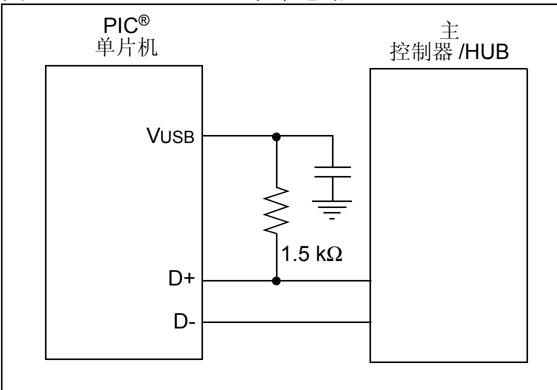
17.2.2.3 内部上拉电阻

PIC18FX455/X550 器件的内部上拉电阻是为了满足低速和全速 USB 的要求而设计的。UPUEN 位（UCFG<4>）使能内部上拉电路。图 17-1 显示了上拉及其控制电路。

17.2.2.4 外部上拉电阻

也可能用到外部上拉电路。可使用 VUSB 引脚上拉 D+ 或 D-。根据 USB 规范的要求，上拉电阻必须为 1.5 kΩ（± 5%）。图 17-3 给出了一个示例。

图 17-3： 外部电路



注： 上图给出了使用片上稳压器和外部上拉电阻的全速配置的典型连接。

17.2.2.5 乒乓缓冲器配置

使用 PPB1:PPB0 位配置乒乓缓冲器。乒乓缓冲器的完整说明请参见**第 17.4.4 节 “乒乓缓冲”**。

17.2.2.6 USB 输出使能监视器

USB 的 $\overline{OE}$ 监视器可以显示 SIE 是正在监听总线还是处于激活状态，正在驱动总线。在使用外部收发器或 UCFG<6> = 1 时，监视功能是默认使能的。

$\overline{OE}$ 监视对于初始系统调试，眼图发生器测试期间的波形触发非常有用。

17.2.2.7 眼图测试使能

当 UCFG<7> 位置 1 时，自动眼图测试可由模块产生。可以根据模块设置来观察眼图输出，这意味着用户要负责配置 SIE 时钟设置、上拉电阻和收发器模式。此外，还必须使能 USB 模块。

一旦 UTEYE 被置 1，模块将仿真从接收状态切换到发送状态并且开始发送 J-K-J-K 位序列（全速模式时为 K-J-K-J 序列）。当使能了眼图测试模式时，此序列将无限重复。

注意当模块连接到实际的 USB 系统时，此位不能被置 1。测试模式专用于电路板检验，有助于进行 USB 认证测试。它旨在向系统开发人员演示 USB 信号的完整噪声，这些噪声来源于电路板布线、阻抗不匹配和其他系统组件。但它不能正确测试从接收到发送状态的转换过程。虽然眼图并不能代替更为复杂的USB认证测试，但是它在第一次系统调试过程中是很有用的。

17.2.2.8 内部稳压器

PIC18FX455/X550 器件内置有 3.3V 稳压器，为内部收发器和内部/外部上拉电路提供电源。为了稳定运行，还需要外接 220 nF （± 20%）电容。

> **注：** VUSB 除了驱动内部收发器之外，还足以驱动一个外部上拉电路。

在默认情况下，稳压器是使能的，并且可以通过 VREGEN 配置位来禁止它。使能时，引脚 VUSB 上会有电压输出。当稳压器被禁止时，必须通过 VUSB 引脚外接 3.3 V 电源为内部收发器供电。如果内部收发器被禁止，则不使用 VUSB。

> **注 1：** 如果 VUSB 接有外部稳压器，就不要使能内部稳压器。
>
> **2：** 即使是禁止了稳压器，V_{DD} 也必须始终大于 VUSB。

17.2.3 USB 状态寄存器 （USTAT）

USB 状态寄存器报告 SIE 中的事务状态。当 SIE 发出 USB 传输完成中断信号时，应该读 USTAT 以确定传输的状态。USTAT 包含传输端点编号、传输方向和乒乓缓冲器指针值 （如果使用的话）。

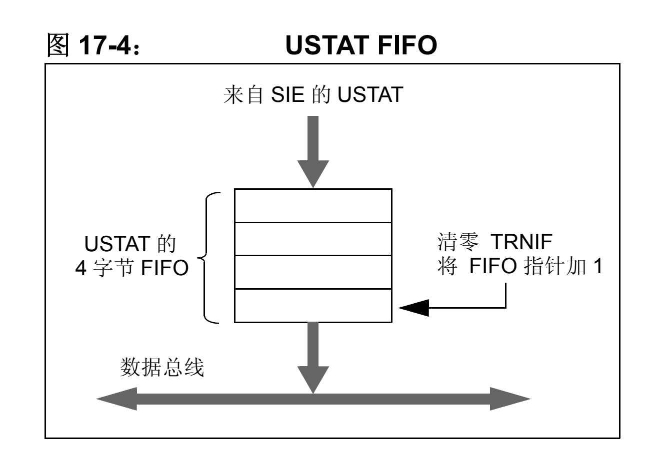
注：	仅在 TRNIF 中断标志位置 1 时，USB 状态寄存器中的数据才有效。

USTAT 寄存器实际上是一个读取由 SIE 维护的 4 个状态字节 FIFO 的窗口。它允许单片机在 SIE 处理其他端点时完成一次传输（图 17-4）。当 SIE 使用缓冲器读写数据结束后，它将更新 USTAT 寄存器。如果在为事务完成中断提供服务之前进行了另一次 USB 传输，SIE 将把下一次传输的状态保存到状态 FIFO 中。

将传输完成标志位 TRNIF 清零，会使 SIE 将 FIFO 指针加 1。如果 FIFO 保持寄存器中的下一个数据有效，SIE 将立即再次产生中断。如果没有其他数据，TRNIF 将保持清零， USTAT 数据将不再可靠。

注：	如果在 USTAT FIFO 已满时接收到一个端点请求， SIE 会自动将一个 NACK 发回给主机。

图 17-4： USTAT FIFO



寄存器 17-3： **USTAT：USB 状态寄存器**

U-0	R-x	R-x	R-x	R-x	R-x	R-x	U-0
—	ENDP3	ENDP2	ENDP1	ENDP0	DIR	PPBI(1)	—
bit 7							bit 0

bit 7 未用：读为 0

bit 6-3 **ENDP3:ENDP0**：上次端点活动的编码 （表示上次 USB 传输更新的 BDT 编号）
1111 = 端点 15
1110 = 端点 14
....
0001 = 端点 1
0000 = 端点 0

bit 2 **DIR**：上次 BD 方向指示位
1 = 上次事务是一个 IN 令牌
0 = 上次事务是一个 OUT 或 SETUP 令牌

bit 1 **PPBI**：乒乓 BD 指针指示位 (1)
1 = 上次事务是针对奇数 BD 存储区进行的
0 = 上次事务是针对偶数 BD 存储区进行的

注 **1**：此位仅对于那些奇偶 BD 寄存器可用的端点有效。

bit 0 未用：读为 0

图注：			
R = 可读位	W = 可写位	U = 未用位，读为 0	
-n = 上电复位时的值	1 = 置 1	0 = 清零	x = 未知

17.2.4 USB 端点控制

16 个可用的双向端点中的每一个都有自己独立的控制寄存器 UEPn （其中 n 表示端点编号）。每个寄存器相应控制位的功能都是相同的，如寄存器 17-4 所示。

EPHSHK 位 （UEPn<4>）控制端点握手，将此位置 1 将使能 USB 握手。通常，除使用同步端点以外，此位总是置 1 的。

EPCONDIS 位 （UEPn<3>）用于使能或禁止通过端点进行的 USB 控制操作 （SETUP）。清零此位将使能 SETUP 事务；注意必须将相应的 EPINEN 和 EPOUTEN 位置 1 以使能 IN 和 OUT 事务。因为 USB 规范中端点 0 是默认控制端点，所以对于端点 0，此位将总是被清零的。

EPOUTEN 位（UEPn<2>）用于使能或禁止主机的 USB OUT 事务。将此位置 1 则使能 OUT 事务。同样，EPINEN 位 （UEPn<1>）将使能或禁止主机的 USB IN 事务。

EPSTALL 位 （UEPn<0>）用于指示端点的 STALL 状态。如果一个特殊的端点发出了一个 STALL，那么 SIE 将把这对端点的 EPSTALL 位置 1。此位将保持置 1 直到由固件清零或 SIE 复位。

寄存器 17-4： UEPn：USB 端点 n 控制寄存器 （UEP0 到 UEP15）

U-0	U-0	U-0	R/W-0	R/W-0	R/W-0	R/W-0	R/W-0
—	—	—	EPHSHK	EPCONDIS	EPOUTEN	EPINEN	EPSTALL[1]
bit 7							bit 0

bit 7-5 未用：读为 0

bit 4 **EPHSHK：**端点握手使能位
1 = 使能端点握手
0 = 禁止端点握手 （通常用于同步端点）

bit 3 **EPCONDIS：**双向端点控制位
如果 EPOUTEN = 1 并且 EPINEN = 1：
1 = 禁止端点 n 的控制传输 ，仅允许 IN 和 OUT 传输
0 = 使能端点 n 的控制 （SETUP）传输 ，也允许 IN 和 OUT 传输

bit 2 **EPOUTEN：**端点输出使能位
1 = 使能端点 n 输出
0 = 禁止端点 n 输出

bit 1 **EPINEN：**端点输入使能位
1 = 使能端点 n 输入
0 = 禁止端点 n 输入

bit 0 **EPSTALL：**端点停止使能位 [1]
1 = 端点 n 停止
0 = 端点 n 不停止

注 1：仅当端点 n 使能时有效；否则，此位可被忽略。

图注：			
R = 可读位	W = 可写位	U = 未用位，读为 0	
-n = 上电复位时的值	1 = 置 1	0 = 清零	x = 未知

17.2.5 USB 地址寄存器 （UADDR）

USB 地址寄存器包含唯一的 USB 地址，当此地址有效时，可被外设识别。当接收到由 URSTIF 表示的 USB 复位信号，或接收到来自单片机的复位信息时，UADDR 复位为 00h。单片机必须在 USB 启动阶段（枚举）写入USB地址，该操作受Microchip USB固件支持。

17.2.6 USB 帧编号寄存器 （UFRMH:UFRML）

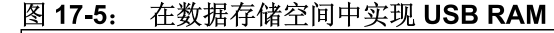
帧编号寄存器包含 11 位帧编号。低字节被保存在 UFRML 中，而高 3 位则被保存在 UFRMH 中。不管是否接收到 SOF 令牌，都用当前帧编号更新该寄存器对。对于单片机而言，这些寄存器是只读的。帧编号寄存器主要用于同步传输。

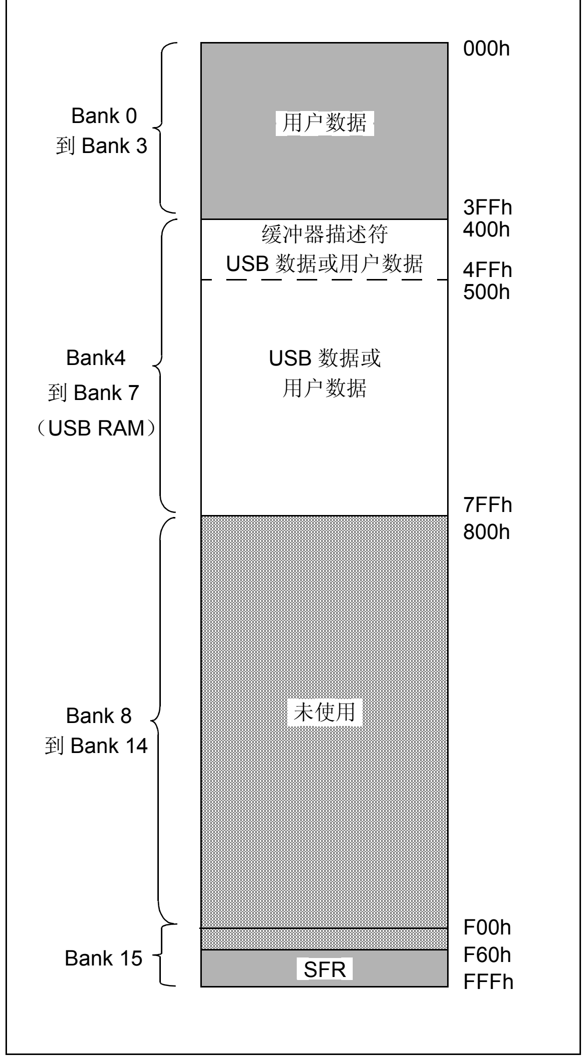
17.3 USB RAM

USB 数据通过称为 USB RAM 的存储空间在单片机内核和 SIE 之间传送。USB RAM 是一个特殊的双端口存储器，被映射到普通数据存储空间的 Bank 4 到 Bank 7 （400h 到 7FFh），总容量为 1 KB （图 17-5）。

Bank 4 （400h 到 4FFh）专用于端点缓冲控制，而 Bank 5 到Bank7 可用于 USB 数据。根据所使用的缓冲类型，除了 Bank 4 的 8 个字节之外，所有字节都可用作 USB 缓冲器空间。

虽然 USB RAM 可用作单片机的数据存储器，但由 SIE 访问的区域不能被单片机访问。这里采用了一种信号机制来确定任意给定时间内对专用缓冲器的访问。这将在**第 17.4.1.1 节 “缓冲器所有权”**中进行讨论。


图 17-5： 在数据存储空间中实现 USB RAM



17.4 缓冲器描述符和缓冲器描述符表

Bank 4 中的寄存器组成称为 “缓冲器描述符表” （Buffer Descriptor Table， BDT）的结构，专门用于控制端点缓冲器。这为用户构建和控制各种长度和配置的端点缓冲器提供了灵活的方法。

BDT 由缓冲描述符（Buffer Descriptor，BD）组成，它用于定义和控制 USB RAM 空间中的实际缓冲器。每个 BD 由四个寄存器组成，其中 n 代表 64 个可能的 BD 中的一个 （范围是 0 到 63）：

- BDnSTAT：BD 状态寄存器
- BDnCNT：BD 字节计数寄存器
- BDnADRL：BD 低地址寄存器
- BDnADRH：BD 高地址寄存器

BD 在序列中总是以一个四字节的数据块 （BDnSTAT:BDnCNT:BDnADRL:BDnADRH）出现， BDnSTAT 的地址总是一个相对于 400h 的偏移量 4n（以十六进制数表示），其中 n 是缓冲器描述符编号。

根据所使用的缓冲配置（**第 17.4.4 节 “乒乓缓冲”**），最多可以有 32、 33 或 64 组缓冲器描述符。BDT 至少要 8 字节长，因为 USB 规范规定，每个设备必须具有初始启动所需的输入和输出端点 0。根据端点和缓冲配置， BDT 可以有 256 个字节。

虽然可以被看作特殊功能控制寄存器，缓冲器描述符的状态和地址寄存器并不是硬件映射的，这与 Bank 15 中的传统单片机 SFR 不同。如果与特定 BD 对应的端点未被使能，则该端点的寄存器将不被使用。它表现为可用的 RAM 空间，而不是未实现的地址。只能通过将 UEPn<1> 位置 1 使能端点，才能将这些地址作为 BD 寄存器使用。如同数据存储区的任何单元一样， BD 寄存器的值在器件复位时都会是不确定的。

图 17-6 所示为一个 BD 的示例，该 BD 用于从 500h 开始的 40 字节缓冲器。如果已经使用 UEPn 寄存器使能了对应的端点，就只有特定的 BD 寄存器组有效。USB RAM 中所有 BD 寄存器都可用。在使能端点之前应该设置好每个端点的 BD。

17.4.1 BD 状态和配置

缓冲器描述符不仅仅定义端点缓冲器的大小，还决定了它的配置和控制。大部分配置都是通过 BD 状态寄存器 BDnSTAT 完成的。每个 BD 都有与自己一一对应的、带编号的 BDnSTAT 寄存器。

图 17-6： 缓冲器描述符示例

	地址	寄存器	内容	
缓冲器描述符	400h	BD0STAT	（xxh）	
	401h	BD0CNT	40h	块大小
	402h	BD0ADRL	00h	起始地址
	403h	BD0ADRH	05h	
缓冲器	500h … 53Fh	USB 数据		

注： 存储区未按比例绘制。

和其他控制寄存器不同， BDnSTAT 寄存器内的位的设置与具体应用息息相关。有两种不同的配置，决定特定时间修改 BD 和缓冲区的是单片机还是 USB 模块。只有一个位是这两种配置共享的。

17.4.1.1 缓冲器所有权

因为 CPU 和 USB 模块共享缓冲器及其 BD，所以使用了简单的信号机制来区分更新 BD 和相关缓冲器的是 CPU 还是 USB。

可以使用 UOWN 位 （BDnSTAT<7>）作为信号来加以区分。

UOWN 是两种配置的 BDnSTAT 中唯一共享的位。

当 UOWN 清零时， BD 由单片机内核 “所有” 。当 UOWN 位置 1 时，BD 和缓冲器模块由 USB 外设 “所有”，此时内核不能够修改 BD 或相应的数据缓冲器。注意，当 SIE 拥有缓冲器时单片机内核仍可以读 BDnSTAT ，反之亦然。

根据寄存器的更新源的不同，缓冲器描述符具有不同的意义。在分配给 USB 外设所有权之前，用户可以通过 BDnSTAT 配置外设的基本操作。此时，也可设置字节计数和缓冲器地址寄存器。

当 UOWN 置 1 时，用户就不能再依赖写入 BD 的值了。此时，SIE 根据需要更新 BD，改写原来的 BD 值。SIE 将使用 PID 令牌来更新 BDnSTAT 寄存器，同时也将更新传输计数值 BDnCNT。

在准备设置端点之前，BDT 的 BDnSTAT 字节总是最后更新的字节。当事务完成时，SIE 将清零 UOWN 位。唯一例外的情况是在使能 KEN 和 / 或 BSTALL 的时候。

当 UOWN 位置 1 时，硬件上并没有任何机制来阻止单片机对存储区的访问。因此，如果单片机尝试在 SIE 拥有 USB RAM 时更改该存储区，可能会出现不可预料的后果。同样，在 USB 外设将 USB RAM 所有权归还给单片机之前，读该存储区也可能会导致数据出错。

17.4.1.2 BDnSTAT 寄存器 （CPU 模式）

当 UOWN = 0 时，单片机内核拥有 BD。此时，该寄存器的其他七个位是控制位。

保持使能位 KEN （BDnSTAT<5>）决定 BD 所有权的归属。如果此位被置 1，一旦 UOWN 位被置 1， BD 将保持被 SIE 拥有，而与端点活动无关。这将防止 USTAT FIFO 被更新，同时防止端点的事务完成中断标志被置 1。只有在选择并行端口而不是 USB RAM 作为数据 I/O 通道时，才使能此功能。

地址递增禁止位 INCDIS （BDnSTAT<4>）控制 SIE 的自动地址递增功能。INCDIS 置 1 将禁止 SIE 在发送或接收每个字节时自动递增缓冲器地址。只有在使用SPP 端口时才使能此功能，此时每个数据字节都被发送到或来自同一个存储单元。

数据 Toggle 同步使能位 DTSEN（BDnSTAT<3>）控制数据传输的奇偶校验。将 DTSEN 置 1 将使能 SIE 的数据 Toggle 同步，将数据包的奇偶校验位对照 DTS （BDnSTAT<6>）的值进行验证。如果包不是同步到达的，数据将被忽略，不会被写入 USB RAM，并且 USB 事务完成中断标志也不会被置 1。SIE 将把 ACK 令牌发回给主机对接收作出应答。表 17-3汇总了DTSEN位对 SIE 的影响。

缓冲器停止位 BSTALL（BDnSTAT<2>）为控制传输提供支持，通常在端点 0 只停止一次。它同时支持 USB 规范第 9 章中指定的 SET_FEATURE/CLEAR_FEATURE 命令，对默认控制端点外的任何端点为特有的连续 STALL。

BSTALL 位使能缓冲器停止。如果接收到的令牌要使用单元中的 BD，BSTALL 置 1 将导致 SIE 将 STALL 令牌返回到主机。相应的 UEPn 控制寄存器中的 EPSTALL 位被置 1，并且在发送 STALL 到主机时产生 STALL 中断。UOWN 位保持置 1 并且 BD 不改变，除非接收到 SETUP 令牌。在这种情况下， STALL 条件被清除并且 BD 的所有权返还给单片机内核。

BD9:BD8 位 （BDnSTAT<1:0>）存储 SIE 字节计数的高 2 位，低 8 位被存储在相应的 BDnCNT 寄存器中。如需更多信息，请参见**第 17.4.2 节 “BD 字节计数”**。

表 17-3： DTSEN 位对奇 / 偶 （DATA0/DATA1）包接收的影响

来自主机的 OUT 包	BDnSTAT 设置		接收数据包之后的设备响应			
	DTSEN	DTS	握手	UOWN	TRNIF	BDnSTAT 和 USTAT 的状态
DATA0	1	0	ACK	0	1	已更新
DATA1	1	0	ACK	1	0	未更新
DATA0	1	1	ACK	0	1	已更新
DATA1	1	1	ACK	1	0	未更新
两者皆有	0	x	ACK	0	1	已更新
两者皆有，有错误	x	x	NAK	1	0	未更新

图注： x = 任意值

寄存器 17-5： BDnSTAT：缓冲器描述符 n 状态寄存器 （BD0STAT 到 BD63STAT），CPU 模式（数据被写入 CPU 控制的存储区）

R/W-x	R/W-x	R/W-x	R/W-x	R/W-x	R/W-x	R/W-x	R/W-x
UOWN （1）	DTS （2）	KEN	INCDIS	DTSEN	BSTALL	BC9	BC8
bit 7							bit 0

bit 7 **UOWN**：USB 所有权位 (1)
0 = 单片机内核拥有 BD 和相应的缓冲器

bit 6 **DTS**：数据 Toggle 同步位 (2)
1 = Data 1 包
0 = Data 0 包

bit 5 **KEN**：BD 保持使能位
1 = 只要 UOWN 置 1， USB 将始终保持 BD （要求 SPP 端点配置）
0 = 只要处理了特定令牌， USB 将归还 BD 的所有权

bit 4 **INCDIS**：地址递增禁止位
1 = 禁止地址递增 （要求 SPP 端点配置）
0 = 使能地址递增

bit 3 **DTSEN**：数据 Toggle 同步使能位
1 = 使能数据 Toggle 同步，具有错误同步值的数据包将被丢弃
0 = 未执行数据 Toggle 同步

bit 2 **BSTALL**：缓冲器停止使能位
1 = 使能缓冲器停止；如果接收到将使用给定单元中 BD 的令牌，将发出 STALL 握手（UOWN 位保持置 1， BD 值不变）
0 = 禁止缓冲器停止

bit 1-0 **BC9:BC8**：字节计数 bit 9 和 bit 8
字节计数位代表在 IN 令牌过程中发送或在 OUT 令牌过程中接收的字节数。和 BC<7:0> 一起使用，有效字节计数的范围为 0-1023。

注 **1**：用户必须在使能 USB 模块之前将此位初始化为需要的值。
2：除非 DTSEN = 1，否则此位被忽略。

图注：			
R = 可读位	W = 可写位	U = 未用位，读为 0	
-n = 上电复位时的值	1 = 置 1	0 = 清零	x = 未知

17.4.1.3 BDnSTAT 寄存器 （SIE 模式）

当 BD 及其缓冲器归 SIE 所有时， BDnSTAT 中的大部分位都有了不同的定义。配置如寄存器 17-6 所示。一旦 UOWN 置 1，以前用户写入的任何数据或控制设置都会被来自 SIE 的数据改写。

SIE 使用令牌包标识符（Packet Identifier， PID）更新 BDnSTAT 寄存器，该 PID 存储在 BDnSTAT<5:3> 中。对应的BDnCNT寄存器中的传输计数被更新，该八位寄存器的值溢出得到的计数值的高两位，被保存在 BDnSTAT<1:0> 中。

17.4.2 BD 字节计数

字节计数代表将在一次 IN 传输过程中被发送的字节总数。在 IN 传输之后，SIE 将返回发送到主机的字节数。

在 OUT 传输过程中，字节计数代表可以被接收并存储到 USB RAM 中的最大字节数。在 OUT 传输之后，SIE 将返回实际接收到的字节数。如果接收到的字节数超过设定的字节计数，数据包将被拒绝，同时还将产生 NAK 握手。发生这种情况时，字节计数不被更新。

10 位字节计数被分别保存到两个寄存器中。低 8 位在 BDnCNT 寄存器中。高 2 位在 BDnSTAT<1:0> 中。因此有效的字节计数范围是 0 到 1023。

17.4.3 BD 地址验证

BD 地址寄存器对包含对应端点缓冲器的起始 RAM 地址。为了让端点起始单元有效，它必须在 USB RAM 中（地址为 400h 到 7FFh）。硬件中没有相应的机制来验证 BD 地址。

如果 BD 地址未指向 USB RAM 中的单元，或者指向另一个端点缓冲器中的地址，数据将很可能丢失或被改写。同样，接收缓冲器（OUT 端点）和正在使用的 BD 单元重叠也将导致不可预料的结果。在开发 USB 应用程序时，用户需要考虑在软件中对地址进行验证。

寄存器 17-6： BDnSTAT：缓冲器描述符 n 状态寄存器 （BD0STAT 到 BD63STAT）， SIE 模式 （数据由 SIE 返回给单片机）

R/W-x	U-x	R/W-x	R/W-x	R/W-x	R/W-x	R/W-x	R/W-x
UOWN	—	PID3	PID2	PID1	PID0	BC9	BC8
bit 7							bit 0

bit 7 **UOWN**：USB 所有权位
1 = SIE 拥有 BD 及其相应的缓冲器

bit 6 **保留**：不被 SIE 写入

bit 5-2 **PID3:PID0**：包标识符位
上一次传输接收到的令牌 PID 值 （仅 IN、 OUT 或 SETUP 事务）。

bit 1-0 **BC9:BC8**：字节计数 bit 9 和 bit 8
这些位由 SIE 更新，反映 OUT 传输时接收的实际字节数和 IN 传输时发送的实际字节数。

图注：			
R = 可读位	W = 可写位	U = 未用位，读为 0	
-n = 上电复位时的值	1 = 置 1	0 = 清零	x = 未知

 初稿

17.4.4 乒乓缓冲

当端点有两组 BD 项时（一组用于偶数数据传输，一组用于奇数数据传输），它被定义为具有乒乓缓冲器。这样就允许在 CPU 处理一组 BD 的同时，SIE 处理另一组 BD。这种方式的乒乓缓冲 BD，可实现单片机与 USB 之间的最大数据吞吐量。

USB 模块支持三种模式的操作：

- 不支持乒乓缓冲
- 仅支持 OUT 端点 0 的乒乓缓冲
- 支持所有端点的乒乓缓冲

使用UCFG寄存器中的PPB1:PPB0位设置乒乓缓冲器。

USB 模块跟踪每个端点的乒乓指针。当使能此模块时，所有指针都复位到指向偶数 BD。 在事务完成之后（UOWN 被 SIE 清零），指针被切换到奇数 BD。在下一次事务完成之后，指针又被切换回偶数 BD。

上次事务的奇偶状态保存在 USTAT 寄存器的 PPBI 位中。用户可以使用 PPBRST 位将所有乒乓指针复位到偶数状态。

图 17-7显示了三种不同的操作模式以及BD是如何填充 USB RAM 的。

根据缓冲配置，BD与特定的端点具有固定的对应关系。表 17-4所示为BD与端点之间映射的详细情况。这种关系也意味着如果没有连续使能端点，BDT 中将出现空白区域。理论上被禁止的端点的 BD 可用作缓冲空间。实际上，用户应该避免使用 BDT 中的这些空间，除非已经采取了验证 BD 地址的措施。

图 17-7： 缓冲模式下的缓冲描述符表映射

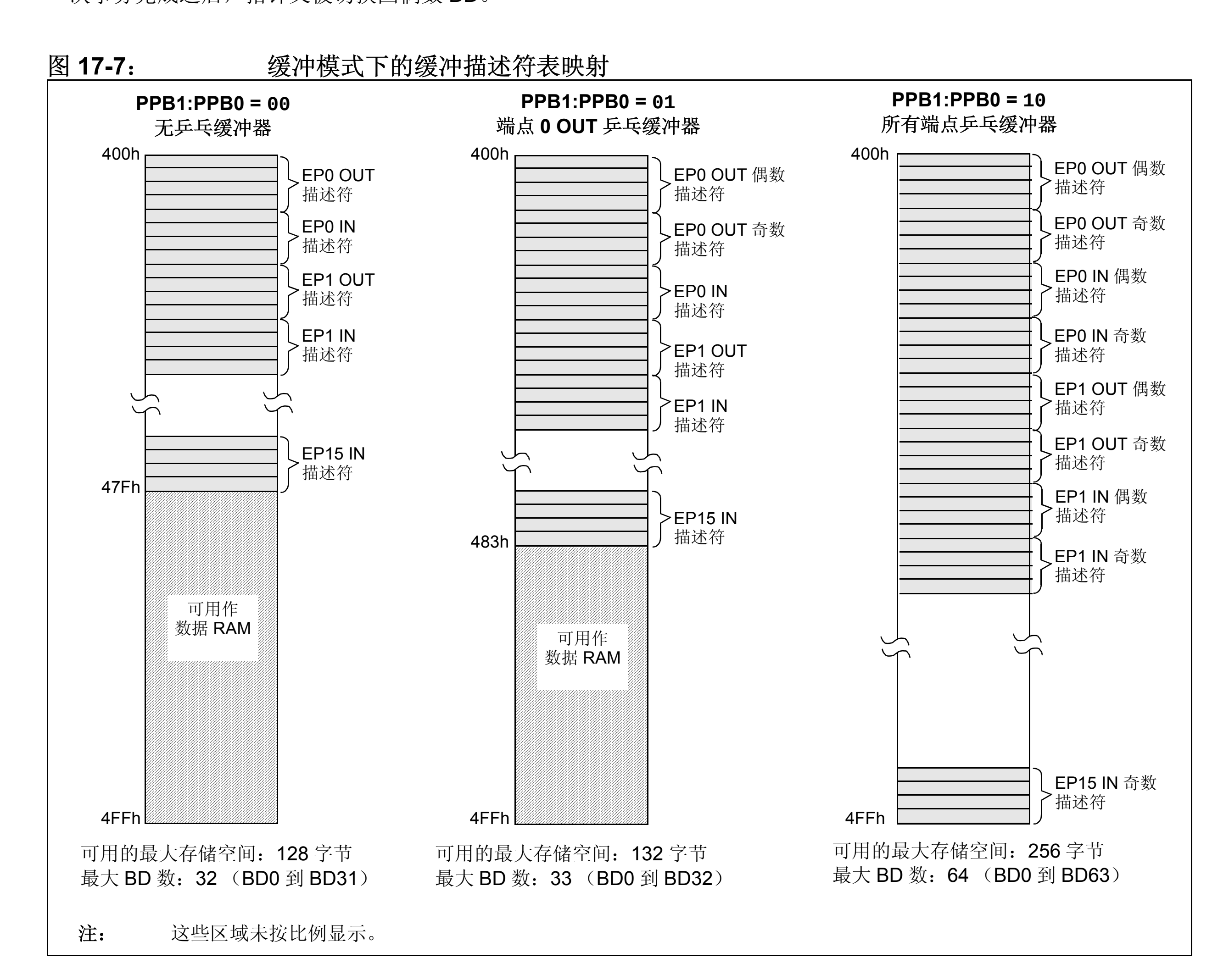


表 17-4： 不同缓冲模式下的缓冲描述符分配

端点	分配给端点的 BD					
	模式 0（无乒乓缓冲）		模式 1（端点 0 OUT 乒乓缓冲）		模式 2（所有端点乒乓缓冲）	
	输出	输入	输出	输入	输出	输入
0	0	1	0 （E），1 （O）	2	0 （E）， 1 （O）	2 （E）， 3 （O）
1	2	3	3	4	4 （E）， 5 （O）	6 （E）， 7 （O）
2	4	5	5	6	8 （E）， 9 （O）	10 （E）， 11 （O）
3	6	7	7	8	12 （E）， 13 （O）	14 （E）， 15 （O）
4	8	9	9	10	16 （E）， 17 （O）	18 （E）， 19 （O）
5	10	11	11	12	20 （E）， 21 （O）	22 （E）， 23 （O）
6	12	13	13	14	24 （E）， 25 （O）	26 （E）， 27 （O）
7	14	15	15	16	28 （E）， 29 （O）	30 （E）， 31 （O）
8	16	17	17	18	32 （E）， 33 （O）	34 （E）， 35 （O）
9	18	19	19	20	36 （E）， 37 （O）	38 （E）， 39 （O）
10	20	21	21	22	40 （E）， 41 （O）	42 （E）， 43 （O）
11	22	23	23	24	44 （E）， 45 （O）	46 （E）， 47 （O）
12	24	25	25	26	48 （E）， 49 （O）	50 （E）， 51 （O）
13	26	27	27	28	52 （E）， 53 （O）	54 （E）， 55 （O）
14	28	29	29	30	56 （E）， 57 （O）	58 （E）， 59 （O）
15	30	31	31	32	60 （E）， 61 （O）	62 （E）， 63 （O）

图注： （E）= 偶数事务缓冲器，（O）= 奇数事务缓冲器

表 17-5： USB 缓冲器描述符表寄存器汇总

名称	Bit 7	Bit 6	Bit 5	Bit 4	Bit 3	Bit 2	Bit 1	Bit 0
BDnSTAT[1]	UOWN	DTS	PID3[2] KEN[3]	PID2[2] INCDIS[3]	PID1[2] DTSEN[3]	PID0[2] BSTALL[3]	BC9	BC8
BDnCNT[1]	字节计数							
BDnADRL[1]	缓冲器地址低字节							
BDnADRH[1]	缓冲器地址高字节							

注 1： 对于缓冲器描述符寄存器， n 的值可以是 0 到 63。为了简洁起见，这 64 个寄存器都显示为一个通用名称代号。所有寄存器都有不确定的复位值 （xxxx xxxx）。

2： 只要寄存器为 SIE 所有 （UOWN 位置 1）， SIE 使用 BDnSTAT 寄存器的 bit 2 到 bit 5 返回 PID3:PID0 值。只要寄存器在 SIE 控制之下，写入 KEN、 INCDIS、 DTSEN 和 BSTALL 的值将不再有效。

3： 在将缓冲器描述符所有权切换到 SIE 之前 （UOWN 位清零）， BDnSTAT 寄存器的 bit 2 到 bit 5 用于配置 KEN、 INCDIS、 DTSEN 和 BSTALL 的值。

 初稿

17.5 USB 中断

USB 模块可以产生多个中断条件。要识别所有中断源，和单片机一样，此模块具有自己的中断逻辑结构。可以使用一组寄存器允许 USB 中断，并用一组单独的标志寄存器来捕获中断事件。在单片机中断逻辑中，所有的 USB 中断源都由一个 USB 中断请求标志位 USBIF （PIR2<5>）来反映。

图 17-8 显示了 USB 模块的中断逻辑。USB 模块中有两层中断寄存器。顶层主要由 USB 状态中断组成，这些中断的允许位和标志位分别在 UIE 和 UIR 寄存器中。第二层由 USB 错误条件中断组成，它们的允许位和标志位分别在 UEIR 和 UEIE 寄存器中。这一层的中断条件都将置位顶层的 USB 错误中断标志 （UERRIF）。

可以使用中断来处理 USB 传输中的常规事务。图 17-9 显示了 USB 帧传输中的某些常见事件和它们相应的中断。

图 17-8： USB 中断逻辑

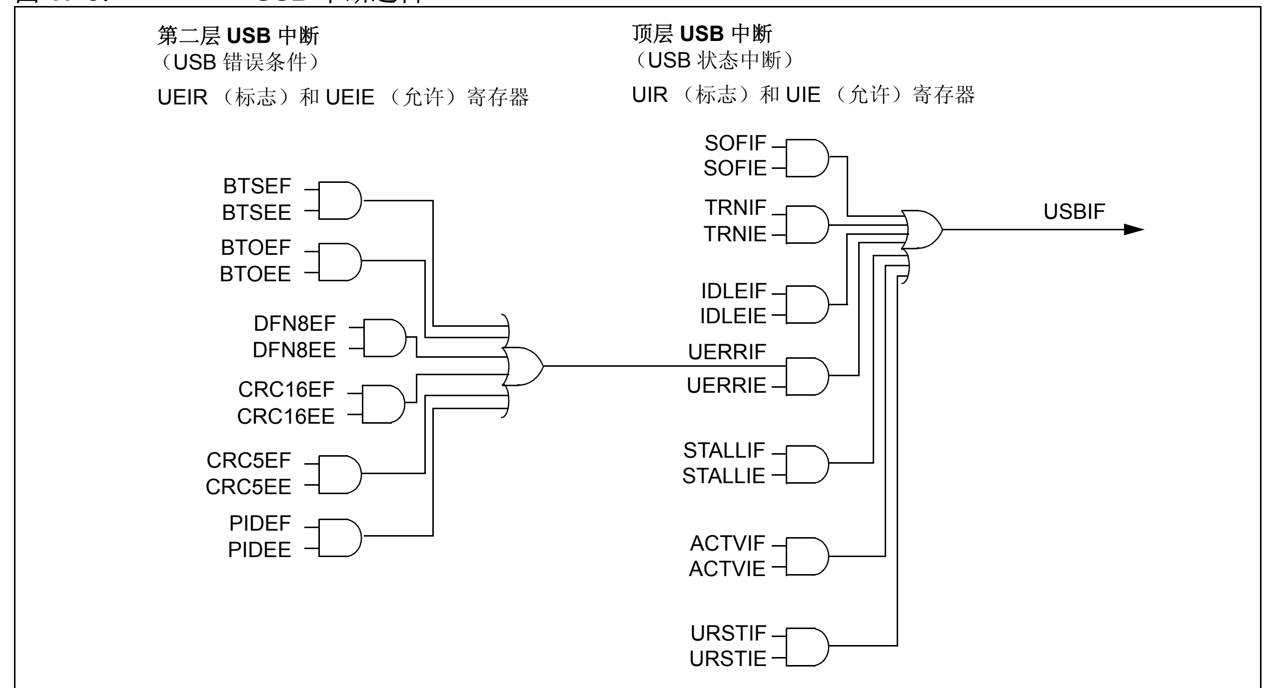
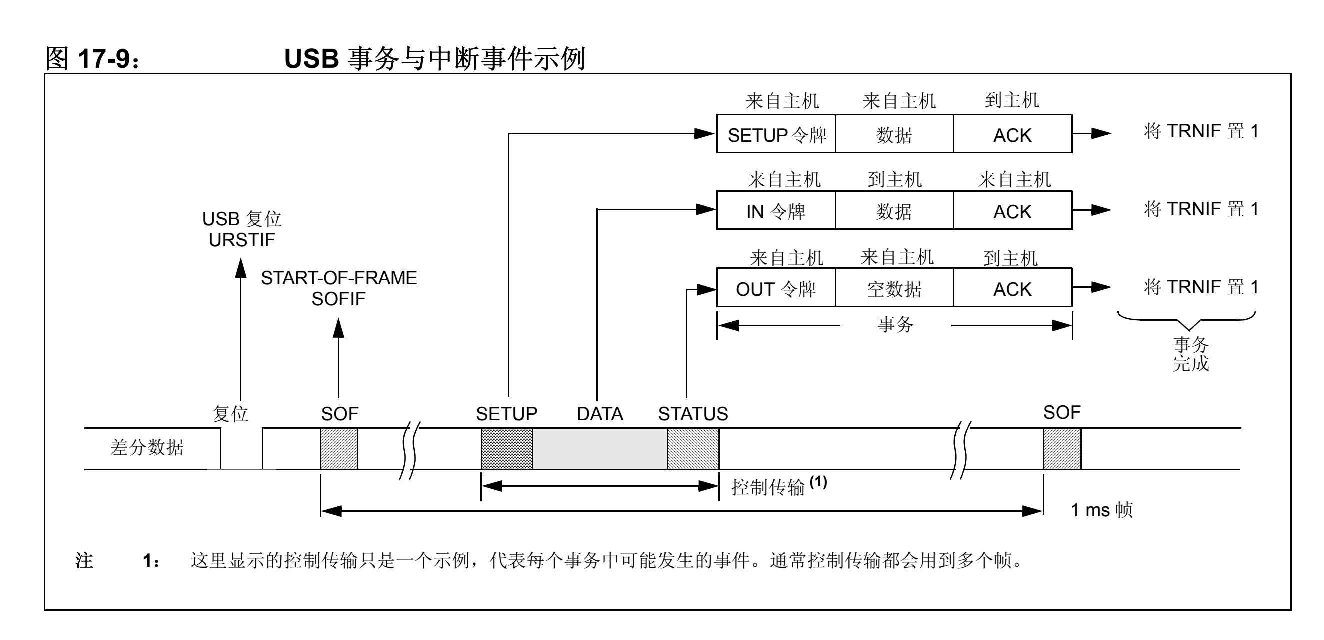


图 17-9： USB 事务与中断事件示例



注 1： 这里显示的控制传输只是一个示例，代表每个事务中可能发生的事件。通常控制传输都会用到多个帧。

17.5.1 USB 中断状态寄存器 （UIR）

USB 中断状态寄存器（寄存器 17-7）包含每个 USB 中断源的状态标志位。这些中断源在 UIE 寄存器中都有对应的中断允许位。所有的 USB 状态标志位一起进行逻辑或运算，产生单片机中断逻辑中的 USBIF 中断标志。

一旦中断标志位被 SIE 置 1，它就必须通过由软件写入 0 来清除。标志位也可以由软件置 1，这有助于固件调试。

寄存器 17-7： UIR：USB 中断状态寄存器

U-0	R/W-0	R/W-0	R/W-0	R/W-0	R/W-0	R-0	R/W-0
—	SOFIF	STALLIF	IDLEIF(1)	TRNIF(2)	ACTVIF(3)	UERRIF(4)	URSTIF
bit 7							bit 0

bit 7 未用：读为 0

bit 6 **SOFIF**：START-OF-FRAME 令牌中断位
1 = SIE 接收到 START-OF-FRAME 令牌
0 = SIE 未接收到 START-OF-FRAME 令牌

bit 5 **STALLIF**：STALL 握手中断位
1 = SIE 发送了 STALL 握手
0 = 未发送 STALL 握手

bit 4 **IDLEIF**：空闲检测中断位 (1)
1 = 检测到空闲条件 （连续空闲时间等于或大于 3 ms）
0 = 未检测到空闲条件

bit 3 **TRNIF**：事务完成中断位 (2)
1 = 待处理的事务处理完成；读 USTAT 寄存器以获取端点信息
0 = 待处理的事务未完成或没有事务等待处理

bit 2 **ACTVIF**：总线活动检测中断位 (3)
1 = 检测到 D+/D- 线上的活动
0 = 未检测到 D+/D- 线上的活动

bit 1 **UERRIF**：USB 错误条件中断位 (4)
1 = 出现未屏蔽的错误条件
0 = 未出现未屏蔽的错误条件

bit 0 **URSTIF**：USB 复位中断位
1 = 发生有效 USB 复位， 00h 被装入 UADDR 寄存器
0 = 未发生 USB 复位

注 **1**：一旦检测到空闲状态，用户通常希望将 USB 模块置于挂起模式。
2：清零此位会导致 USTAT FIFO 指针加 1（仅对于 IN、OUT 和 SETUP 令牌有效）。
3：此位通常只有在检测到 UIDLE 中断事件后才不被屏蔽。
4：只有通过 UEIE 寄存器允许错误条件中断才能将此位置 1。此位仅仅是一个状态位，并且不能被用户置 1 或清零。

图注：			
R = 可读位	W = 可写位	U = 未用位，读为 0	
-n = 上电复位时的值	1 = 置 1	0 = 清零	x = 未知

 初稿

17.5.2 USB 中断允许寄存器 （UIE）

USB 中断允许寄存器（寄存器 17-8）包含 USB 状态中断源的允许位。将这些位置 1 将分别允许 UIR 寄存器中的中断源。

此寄存器中的值仅影响中断条件是否会被传递给单片机中断逻辑。中断条件会将标志位置 1，从而可以对它们进行查询和在不产生实际中断的情况下提供中断服务。

寄存器 17-8： UIE：USB 中断允许寄存器

U-0	R/W-0	R/W-0	R/W-0	R/W-0	R/W-0	R/W-0	R/W-0
—	SOFIE	STALLIE	IDLEIE	TRNIE	ACTVIE	UERRIE	URSTIE
bit 7							bit 0

bit 7 未用：读为 0

bit 6 **SOFIE**：START-OF-FRAME 令牌中断允许位
1 = 允许 START-OF-FRAME 令牌中断
0 = 禁止 START-OF-FRAME 令牌中断

bit 5 **STALLIE**：STALL 握手中断允许位
1 = 允许 STALL 中断
0 = 禁止 STALL 中断

bit 4 **IDLEIF**：空闲检测中断允许位
1 = 允许空闲检测中断
0 = 禁止空闲检测中断

bit 3 **TRNIE**：事务完成中断允许位
1 = 允许事务中断
0 = 禁止事务中断

bit 2 **ACTVIE**：总线活动检测中断允许位
1 = 允许总线活动检测中断
0 = 禁止总线活动检测中断

bit 1 **UERRIE**：USB 错误中断允许位
1 = 允许 USB 错误中断
0 = 禁止 USB 错误中断

bit 0 **URSTIE**：USB 复位中断允许位
1 = 允许 USB 复位中断
0 = 禁止 USB 复位中断

图注：			
R = 可读位	W = 可写位	U = 未用位，读为 0	
-n = 上电复位时的值	1 = 置 1	0 = 清零	x = 未知

17.5.3 USB 错误中断状态寄存器 （UEIR）

USB 错误中断状态寄存器（寄存器 17-9）包含 USB 外设每个错误源的标志位。这些中断源在 UEIE 寄存器中都有对应的中断允许位。所有的 USB 错误标志位一起进行逻辑或运算，在中断逻辑的顶层产生 USB 错误中断标志 （UERRIF）。

在检测到错误条件时，会立即将对应错误位置 1。因此，中断通常不会在令牌处理结束时产生。

一旦 SIE 将中断允许位置 1，就必须由软件写 0 才能将它清零。

寄存器 17-9： UEIR：USB 错误中断状态寄存器

R/C-0	U-0	U-0	R/C-0	R/C-0	R/C-0	R/C-0	R/C-0
BTSEF	—	—	BTOEF	DFN8EF	CRC16EF	CRC5EF	PIDEF
bit 7							bit 0

bit 7 **BTSEF**：位填充错误标志位
1 = 检测到位填充错误
0 = 无位填充错误

bit 6-5 未用：读为 0

bit 4 **BTOEF**：总线周转 （Turn-around）超时错误标志位
1 = 发生总线周转超时 （在上一次 EOP 之后，已发生了 16 比特以上的空闲时间）
0 = 未发生总线周转超时

bit 3 **DFN8EF**：数据字段大小错误标志位
1 = 数据字段的字节数不是整数
0 = 数据字段的字节数是整数

bit 2 **CRC16EF**：CRC16 失败标志位
1 = CRC16 失败
0 = CRC16 通过

bit 1 **CRC5EF**：CRC5 主机错误标志位
1 = 令牌包由于 CRC5 错误被拒绝
0 = 令牌包被接收

bit 0 **PIDEF**：PID 检查失败标志位
1 = PID 检查失败
0 = PID 检查通过

图注：			
R = 可读位	C = 可清零位	U = 未用位，读为 0	
-n = 上电复位时的值	1 = 置 1	0 = 清零	x = 未知

17.5.4 USB 错误中断允许寄存器 （UEIE）

USB 错误中断允许寄存器 （寄存器 17-10）包含 USB 错误中断源的允许位。将这些位中的任何位置 1，都将把 UEIR 寄存器中相应的错误中断源传递给中断逻辑顶层的 UERR 位。

同 UIE 寄存器一样，上述允许位仅影响中断条件是否会被传递给单片机中断逻辑。中断条件仍会将标志位置 1，从而允许对标志位进行查询或在并未产生实际中断的情况下提供中断服务。

寄存器 17-10： UEIE：USB 错误中断允许寄存器

R/W-0	U-0	U-0	R/W-0	R/W-0	R/W-0	R/W-0	R/W-0
BTSEE	—	—	BTOEE	DFN8EE	CRC16EE	CRC5EE	PIDEE
bit 7							bit 0

bit 7 **BTSEE：**位填充错误中断允许位
1 = 允许位填充错误中断
0 = 禁止位填充错误中断

bit 6-5 未用：读为 0

bit 4 **BTOEE：**总线周转超时错误中断允许位
1 = 允许总线周转超时错误中断
0 = 禁止总线周转超时错误中断

bit 3 **DFN8EE：**数据字段大小错误中断允许位
1 = 允许数据字段大小错误中断
0 = 禁止数据字段大小错误中断

bit 2 **CRC16EF：**CRC16 失败中断允许位
1 = 允许 CRC16 失败中断
0 = 禁止 CRC16 失败中断

bit 1 **CRC5EE：**CRC5 主机错误中断允许位
1 = 允许 CRC5 主机错误中断
0 = 禁止 CRC5 主机错误中断

bit 0 **PIDEE：**PID 检查失败中断允许位
1 = 允许 PID 检查失败中断
0 = 禁止 PID 检查失败中断

图注：			
R = 可读位	W = 可写位	U = 未用位，读为 0	
-n = 上电复位时的值	1 = 置 1	0 = 清零	x = 未知

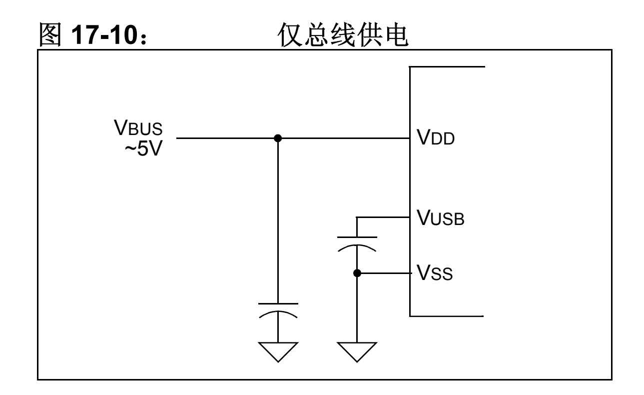
17.6 USB 电源模式

很多 USB 应用都有几种不同的电源需求和配置。最常见的电源模式有仅总线供电（Bus Power Only）、仅自供电（Self-Power Only）和以自供电为主的双电源（Dual Power）供电模式。这里列出了大部分常见情形。

17.6.1 仅总线供电模式

在仅总线供电的模式下，应用的所有电源均来自 USB（图 17-10）。这是最简单的设备供电方式。

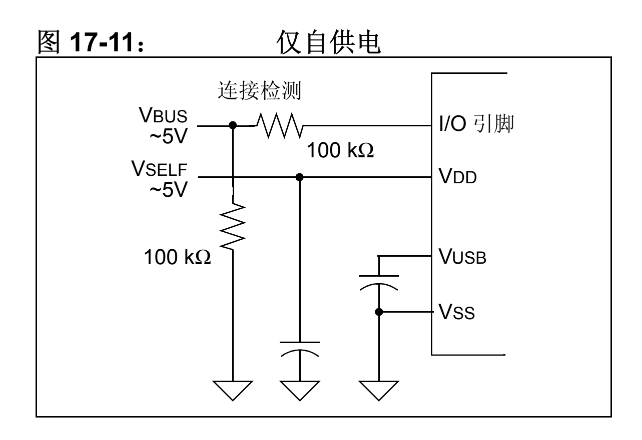
图 17-10： 仅总线供电



17.6.2 仅自供电模式

在仅自供电的模式下，USB 应用为自身提供电源，只有很小一部分电源来自USB，如图 17-11 所示。注意该示例加上了一个连接指示器，用于显示何时连接上 USB。

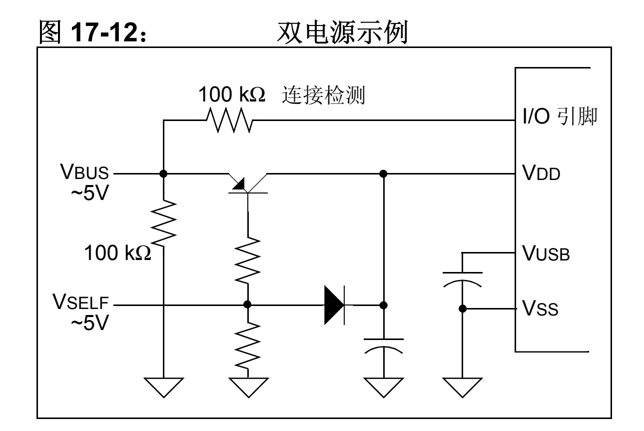
图 17-11： 仅自供电



17.6.3 以自供电为主的双电源供电

某些应用可能需要双电源供电。在这种供电模式下，应用主要使用内部电源，但是在没有内部电源可用的时候切换到 USB 电源。图 17-12 给出了以自供电为主的双电源供电的简单示例，它可以在仅自供电和仅 USB 总线供电的模式之间自动切换。

图 17-12： 双电源示例



注：	用户应该牢记由 USB 总线供电时的电流限制。根据 USB 规范 2.0，对于低功耗设备，不能超过 100 mA，而对于高功耗设备不能超过 500 mA。

17.7 并行通信端口

并行通信端口（SPP）是除 USB RAM 之外用于数据传输的备用通道。可以将端点配置为使用 SPP 将数据直接发送到外部硬件或直接接收来自外部硬件的数据。

此方法可以把单片机作为数据管理器，直接通过SPP传递大块的数据而无需单片机进行处理。应用示例将包含数据采集系统，其中外部 FIFO 的数据通过 USB 和主机进行通信。在这种情况下，单片机控制端点，而原始数据的传送则靠外设完成。

通过相关的端点缓冲器描述符将 SPP 使能为 USB 端点的端口。必须通过以下方法来使能端点：

1. 将 BDnADRL:BDnADRH 设置为指向 FFFFh。
2. 将 KEN 位（BDnSTAT<5>）置 1 以便让 SIE 保持对缓冲器的控制。
3. 将 INCDIS 位（BDnSTAT<4>）置 1 以禁止地址自动递增。

欲知更多有关 SPP 的信息，请参见**第 18.0 节“并行通信端口”**。

注	1：	如果端点被配置为使用 SPP，就必须将 SPP 模块配置为使用 USB 模块。否则，将会导致不可预料的操作。
	2：	此外，如果将端点配置为使用 SPP，该端点的数据传输类型只能为同步传输。

17.8 振荡器

USB 模块有特定的时钟要求。在全速模式下，其时钟源必须为 48 MHz。即便如此，并不要求单片机内核和其他外设以此时钟速度运行或使用同一个时钟源。第 **2.3** 节 **"USB 的振荡器设置"**中详细描述了可用的时钟选项。

17.9 USB 固件和驱动程序

Microchip 提供了一些针对应用的特定的资源，比如 USB 固件和驱动程序。最新的固件和驱动程序请参见 www.microchip.com。

表 17-6： 与 USB 模块操作相关的寄存器 (1)

名称	Bit 7	Bit 6	Bit 5	Bit 4	Bit 3	Bit 2	Bit 1	Bit 0	详情请见：（页）
INTCON	GIE/GIEH	PEIE/GIEL	TMR0IE	INT0IE	RBIE	TMR0IF	INT0IF	RBIF	51
IPR2	OSCFIP	CMIP	USBIP	EEIP	BCLIP	HLVDIP	TMR3IP	CCP2IP	54
PIR2	OSCFIF	CMIF	USBIF	EEIF	BCLIF	HLVDIF	TMR3IF	CCP2IF	54
PIE2	OSCFIE	CMIE	USBIE	EEIE	BCLIE	HLVDIE	TMR3IE	CCP2IE	54
UCON	—	PPBRST	SE0	PKTDIS	USBEN	RESUME	SUSPND	—	55
UCFG	UTEYE	UOEMON	—	UPUEN	UTRDIS	FSEN	PPB1	PPB0	55
USTAT	—	ENDP3	ENDP2	ENDP1	ENDP0	DIR	PPBI	—	55
UADDR	—	ADDR6	ADDR5	ADDR4	ADDR3	ADDR2	ADDR1	ADDR0	55
UFRML	FRM7	FRM6	FRM5	FRM4	FRM3	FRM2	FRM1	FRM0	55
UFRMH	—	—	—	—	—	FRM10	FRM9	FRM8	55
UIR	—	SOFIF	STALLIF	IDLEIF	TRNIF	ACTVIF	UERRIF	URSTIF	55
UIE	—	SOFIE	STALLIE	IDLEIE	TRNIE	ACTVIE	UERRIE	URSTIE	55
UEIR	BTSEF	—	—	BTOEF	DFN8EF	CRC16EF	CRC5EF	PIDEF	55
UEIE	BTSEE	—	—	BTOEE	DFN8EE	CRC16EE	CRC5EE	PIDEE	55
UEP0	—	—	—	EPHSHK	EPCONDIS	EPOUTEN	EPINEN	EPSTALL	55
UEP1	—	—	—	EPHSHK	EPCONDIS	EPOUTEN	EPINEN	EPSTALL	55
UEP2	—	—	—	EPHSHK	EPCONDIS	EPOUTEN	EPINEN	EPSTALL	55
UEP3	—	—	—	EPHSHK	EPCONDIS	EPOUTEN	EPINEN	EPSTALL	55
UEP4	—	—	—	EPHSHK	EPCONDIS	EPOUTEN	EPINEN	EPSTALL	55
UEP5	—	—	—	EPHSHK	EPCONDIS	EPOUTEN	EPINEN	EPSTALL	55
UEP6	—	—	—	EPHSHK	EPCONDIS	EPOUTEN	EPINEN	EPSTALL	55
UEP7	—	—	—	EPHSHK	EPCONDIS	EPOUTEN	EPINEN	EPSTALL	55
UEP8	—	—	—	EPHSHK	EPCONDIS	EPOUTEN	EPINEN	EPSTALL	55
UEP9	—	—	—	EPHSHK	EPCONDIS	EPOUTEN	EPINEN	EPSTALL	55
UEP10	—	—	—	EPHSHK	EPCONDIS	EPOUTEN	EPINEN	EPSTALL	55
UEP11	—	—	—	EPHSHK	EPCONDIS	EPOUTEN	EPINEN	EPSTALL	55
UEP12	—	—	—	EPHSHK	EPCONDIS	EPOUTEN	EPINEN	EPSTALL	55
UEP13	—	—	—	EPHSHK	EPCONDIS	EPOUTEN	EPINEN	EPSTALL	55
UEP14	—	—	—	EPHSHK	EPCONDIS	EPOUTEN	EPINEN	EPSTALL	55
UEP15	—	—	—	EPHSHK	EPCONDIS	EPOUTEN	EPINEN	EPSTALL	55

图注： — = 未用（读为 0）。USB 模块未使用阴影单元。

注 1： 此表仅包含数据存储空间 Bank 15 中的硬件映射的 SFR。映射到 Bank 4 中的缓冲器描述符寄存器不是真正的 SFR，这些寄存器在表 17-5 中已单独列出。

17.10 USB 概述

本节讨论了一些基本的 USB 概念和设计 USB 设备所需的有用知识。虽然本节涵盖了很多内容，但是 USB 规范及其分类规范包含了更为详尽的信息。因此，如需了解更多信息，建议读者参考 USB 规范（www.usb.org）。如果您非常熟悉 USB，阅读本节可以帮助您快速重温 USB 的相关知识。

17.10.1 分层架构

USB 设备的功能是通过构建一个分层框架实现的，如图 17-13 所示。每层都与设备的某一功能相关。最高层是配置层。设备可以有多种配置，例如，某种设备可能在仅自供电和仅总线供电模式的基础上有多种电源要求。

对于一种配置，可能有多个接口。每个接口都支持该配置的一种特定模式。

接口下面就是端点。数据直接在该层传送。最多可有 16 个双向端点。端点 0 总是一个控制端点，并且在默认情况下端点 0 必须可用于配置总线中的设备。

17.10.2 帧

总线上的通信信息被划分为多个 1 ms 的时隙，称为帧。每帧都包含多个设备和端点的多种事务。图 17-9 所示为一个帧内的事务示例。

17.10.3 传输

USB 规范中定义了四种传输类型：

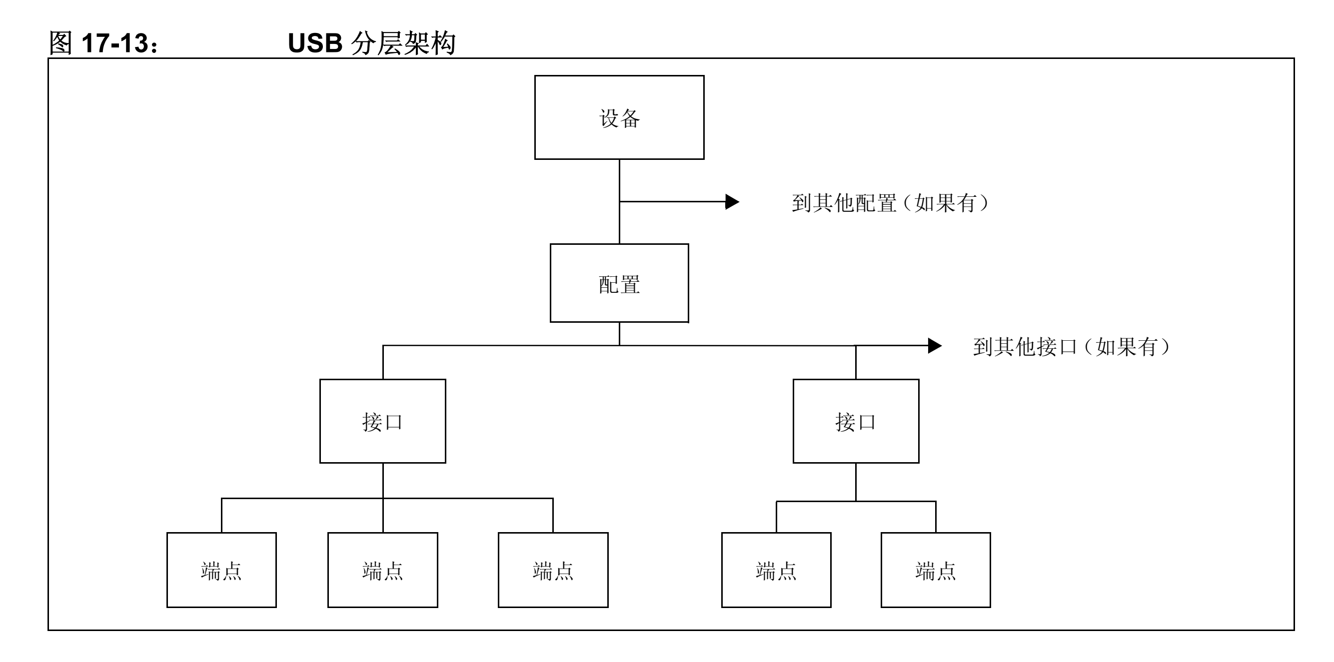
- **同步：**这种传输类型可确保及时传递大量数据（最大为 1023 字节），但是不能保证数据的完整性。它适用于丢失少量数据无关紧要的通信应用，比如音频数据。
- **批量：**这种传输类型允许在传输大量数据的同时确保数据的完整性，但是传输时限不能保证。
- **中断：**这种传输类型可以确保及时传输少量数据，并确保数据的完整性。
- **控制：**这种类型提供设备启动控制。

全速设备支持所有传输类型，而低速设备只能使用中断和控制传输方式。

17.10.4 电源

可使用 USB 供电。USB 规范定义了总线供电要求。设备可以由自身供电也可以由总线供电。自供电的设备的电源来自外部电源，而总线供电的设备电源来自总线。

图 17-13： USB 分层架构



USB 规范中对取自总线的电源功率作出了限制。在电压大约为 5V 的情况下，每个设备可确保 100 mA 的电流（一个单位负载）。可以要求的最大电流为 500 mA。注意，可以请求超过一个单位负载的电源功率，但主机或 hub 不一定会提供额外的电流。因此，需要消耗一个单位负载以上功耗的设备必须确保具有一个单位负载或更低的低功耗配置。

USB 规范也定义了挂起模式。在这种情况下，1 秒钟内的平均电流必须被限制在 500 µA 以内。一旦无活动状态保持 3 ms （即 3 ms 没有 SOF 令牌）之后，设备必须进入挂起状态。进入挂起状态的设备必须在挂起后 10 ms 之内降低电流消耗。同样，在唤醒时，设备必须在 10 ms 之内将电流升高到挂起门限值之上以示唤醒。

17.10.5 枚举

当设备最初连接到总线时，总线进入枚举过程，尝试识别设备。基本上，主机会询问设备，收集一些信息，比如功耗、数据传输速率和大小、协议和其他描述信息；描述符就包含这些信息。枚举过程通常包括以下事件：

1. USB 复位：复位设备。因此，设备未被配置并且没有地址 （地址 0）。
2. 获取设备描述符：主机请求设备描述符的一小部分。
3. USB 复位：再次复位设备。
4. 设置地址：主机为设备分配地址。
5. 获取设备描述符：主机再次获取设备描述符，收集信息，如制造商、设备类型以及最大控制包大小等。
6. 获取配置描述符。
7. 获取其他描述符。
8. 进行配置。

具体的枚举过程由主机决定。

17.10.6 描述符

共有 8 种标准描述符类型，其中的 5 种对设备来讲非常重要。

17.10.6.1 设备描述符

设备描述符提供一般信息，比如制造商、产品编号、序列号、设备类别和配置数。只有一个设备描述符。

17.10.6.2 配置描述符

配置描述符提供设备的电源要求，以及此配置所支持的接口数的信息。一个设备有多种配置（即低功耗配置和高功耗配置）。

17.10.6.3 接口描述符

接口描述符提供了此接口所使用的端点数量和接口类别的信息。一个配置可能有多个接口。

17.10.6.4 端点描述符

端点描述符确定传输类型 （见第 **17.10.3 节 “传输”**）和方向以及端点的其他信息。一个设备可能有多个端点并且它们可以共享多种不同配置。

17.10.6.5 字符串描述符

前面的多种描述符都引用了一个或多个字符串描述符。字符串描述符提供了与层 （见第 **17.10.1 节 “分层架构”**）相关的可读信息。通常这些字符串在主机中，用来帮助用户识别设备。字符串描述符通常是可选的，以节省存储空间，并且采用统一的编码格式。

17.10.7 总线速度

每个 USB 设备都必须告知主机其总线状态及速度。这是在连接时通过一个连到总线的 1.5 kΩ 电阻完成的。

根据设备的速度，电阻将 D+ 或 D- 线上拉到 3.3V。低速设备的上拉电阻连接到 D- 线。全速设备的上拉电阻连接到 D+ 线。

17.10.8 分类规范和驱动程序

USB 规范包括操作系统开发商可选择支持的分类规范。类别包括音频、大容量存贮器、通信和人机交互（Human Interface， HID）。在大部分情况下，主机方需要驱动程序来与 USB 设备进行通信。因此在客户应用中，需要开发驱动程序。幸运的是，对于大部分常见类型的设备来说，大部分常见主机系统中均配有驱动程序。因此，这些驱动程序可以被重复使用。

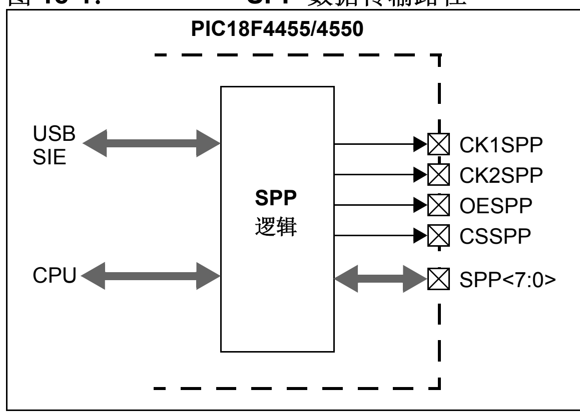
注：

18.0 并行通信端口

注：	仅 40/44 引脚器件上有并行通信端口。

PIC18F4455/4550 USB 器件提供一个并行通信端口作为与外部系统传输数据的高速接口。此并行端口作为一个主控端口连同片选和时钟输出来控制与从器件的数据传输。数据可以被直接传输到USB SIE 或被传输到单片机内核。图 18-1 显示了 SPP 数据传输路径的框图。

图 18-1： SPP 数据传输路径



此外，SPP 通过一个选通输出线可以提供时分复用寻址信息。因此，可以同时写入该 USB 端点的数据及其端点号。

18.1 SPP 配置

有两个用于控制 SPP 工作方式的寄存器：SPPCON 和 SPPCFG。SPPCON 寄存器（寄存器 18-1）全面控制并行端口的工作并决定它受USB控制还是单片机控制。SPPCFG 寄存器（寄存器 18-2）控制时序设置和引脚输出。

18.1.1 使能 SPP

将 SPPEN 位 （SPPCON<0>）置 1 来使能 SPP。另外，必须对与 SPP 引脚相对应的 TRIS 位进行适当的配置。至少：

- TRISD<7:0> 位必须置 1 （= 1）
- TRISE<2:1> 位必须清零 （= 0）

如果使用 CK1SPP：

- TRISE<0> 位必须清零 （= 0）

如果使用 CSPP：

- TRISB<4> 位必须清零 （= 0）

寄存器 18-1： SPPCON：SPP 控制寄存器

U-0	U-0	U-0	U-0	U-0	U-0	R/W-0	R/W-0
—	—	—	—	—	—	SPPOWN	SPPEN
bit 7							bit 0

bit 7-2 未用：读为 0

bit 1 **SPPOWN**：SPP 控制所有权位
1 = USB 外设控制 SPP
0 = 单片机直接控制 SPP

bit 0 **SPPEN**：SPP 使能位
1 = 使能 SPP
0 = 禁止 SPP

图注：			
R = 可读位	W = 可写位	U = 未用位，读为 0	
-n = 上电复位时的值	1 = 置 1	0 = 清零	x = 未知

 初稿

寄存器 18-2： SPPCFG：SPP 配置寄存器

R/W-0	R/W-0	R/W-0	R/W-0	R/W-0	R/W-0	R/W-0	R/W-0
CLKCFG1	CLKCFG0	CSEN	CLK1EN	WS3	WS2	WS1	WS0
bit 7							bit 0

bit 7-6 **CLKCFG1:CLKCFG0**：SPP 时钟配置位

`1x` = 当读到或写入一个奇数端点地址时， CLK1 电平翻转。
当读到或写入一个偶数端点地址时， CLK2 电平翻转。
`01` = 写时 CLK1 电平翻转；读时 CLK2 电平翻转
`00` = 只有在写端点地址时 CLK1 电平才翻转；在读或写数据时 CLK2 电平翻转

bit 5 **CSEN**：SPP 片选引脚使能位

`1` = RB4 引脚受控于 SPP 模块并作为 SPP CS 输出
`0` = RB4 作为数字 I/O 端口

bit 4 **CLK1EN**：SPP CLK1 引脚使能位

`1` = RE0 引脚受控于 SPP 模块并作为 SPP CLK1 输出
`0` = RE0 作为数字 I/O 端口

bit 3-0 **WS3:WS0**：SPP 等待状态位

`1111` = 30 个额外等待状态
`1110` = 28 个额外等待状态
• •
• •
`0001` = 2 个 额外等待状态
`0000` = 0 个 额外等待状态

图注：			
R = 可读位	W = 可写位	U = 未用位，读为 0	
-n = 上电复位时的值	1 = 置 1	0 = 清零	x = 未知

18.1.2 时钟数据

SPP 有四路输出控制信号：

- 两个独立的时钟输出 （CK1SPP 和 CK2SPP）
- 输出使能 （OESPP）
- 片选 （CSSPP）

它们组合起来有几种不同的配置，可用于控制到从器件的数据流。当使用所有控制输出时，有三个主要的选项：

- CLK1 为传输端点地址提供时钟， CLK2 为数据传输提供时钟
- CLK1 为写操作提供时钟， CLK2 为读操作提供时钟
- CLK1 为传输奇地址数据提供时钟， CLK2 为传输偶地址数据提供时钟

其他控制选项需要禁止 CK1SPP 和 CSSPP 输出才能使用。寄存器 18-2 中的 CLK1EN 和 CSEN 位可分别使能或禁止这些控制选项。

18.1.3 等待状态

SPP还具有把等待状态添加到读或写操作的功能。这样可以访问那些需要较长存取时间的并行器件。

等待状态时钟由数据源时钟产生。若 SPP 被配置为 USB 端点，那么等待状态时钟将由 USB 时钟产生。同样，若 SPP 被配置为在单片机控制下运行，那么等待状态时钟将由指令时钟 （Fosc/4）产生。

WS3:WS0 位用于设置 SPP 使用的等待状态，范围从无等待状态到 30 个等待状态，以 2 为单位递增。等待状态会对称地添加到所有事件，每经过一个时钟周期 （正常情况下，一次需用两个时钟周期）就会添加半数的等待状态。图 18-3 和图 18-4 显示了为每个事件添加 4 个等待状态的信号实例。

18.1.4 SPP 上拉电路

SPP 数据线 （SPP<7:0>）配备有内部上拉电路，适用于那些端口处于高阻态的应用。上拉电路可以使用控制位 RDPU （PORTE<7>）使能。

图 **18-2：** 单片机写地址、写数据和读数据时序 （无等待状态）

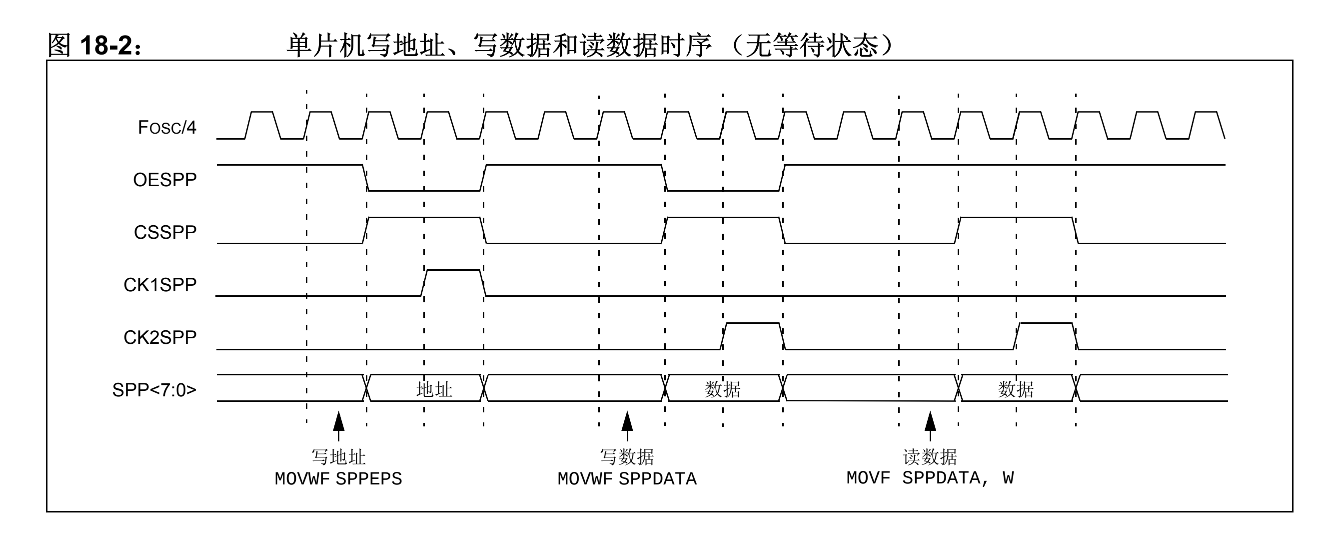


图 **18-3：** **USB** 写地址和数据时序 （**4** 个等待状态）

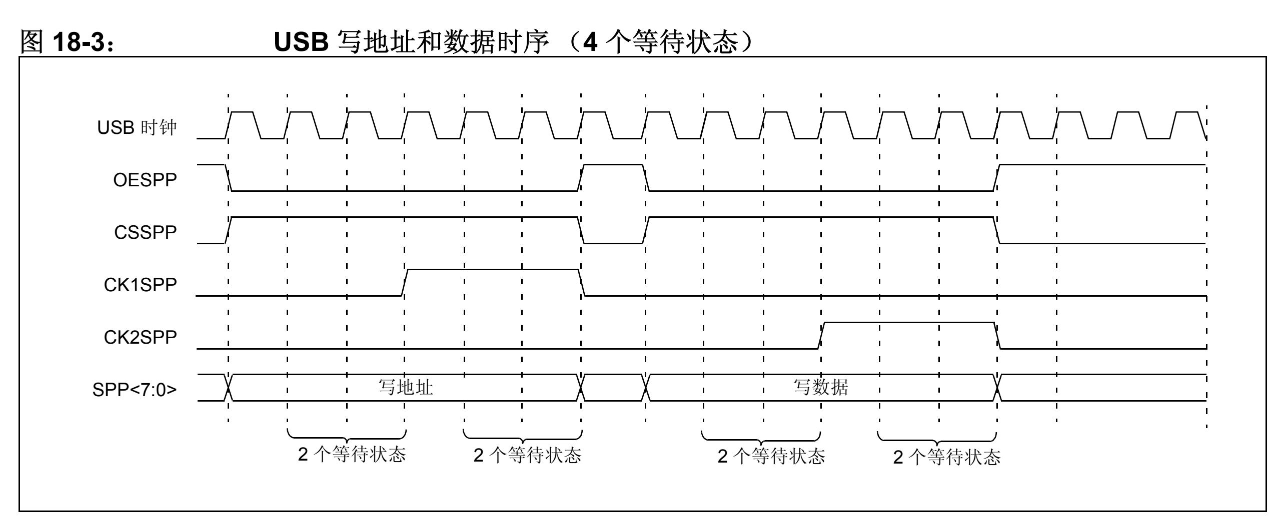
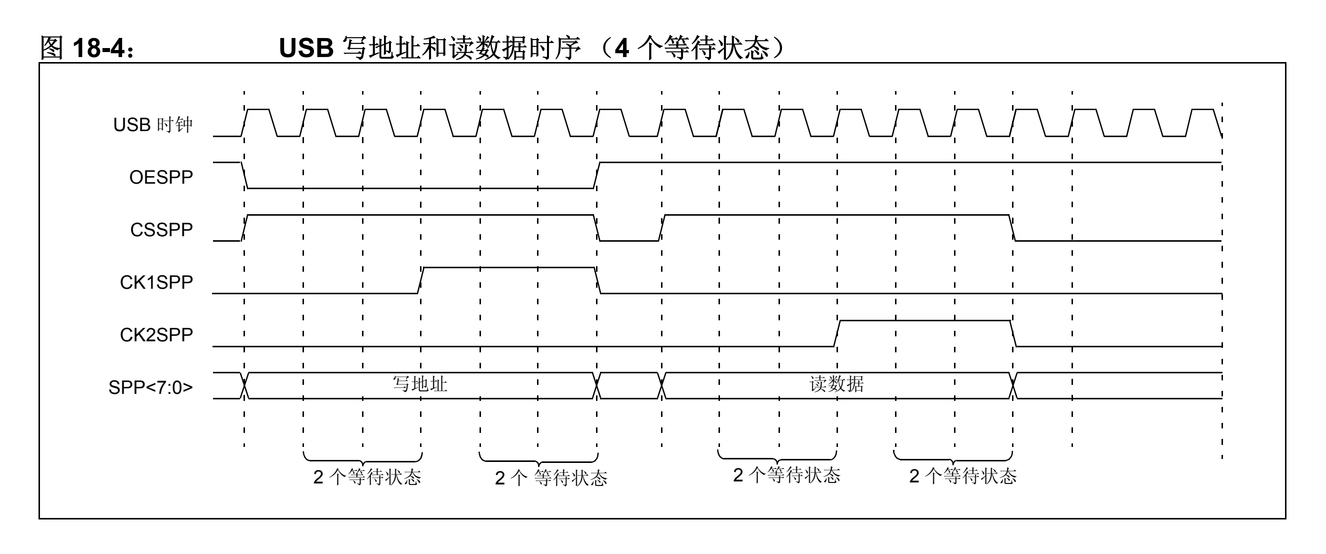


图 **18-4：** **USB** 写地址和读数据时序 （**4** 个等待状态）



18.2 USB 控制设置

当 SPP 被配置为用于 USB 操作时，可直接在时钟控制下与 USB 外设进行数据通信而无需单片机的参与。因此，不占用 CPU 处理时间。如图 18-5 所示，数据在时钟控制下从 SPP 输入或输出，首先传输的是端点（地址）信息，紧接着是一个或多个数据字节。这种方式特别适用于同步传输大量数据的应用。

要设置 SPP 使之受 USB 控制需要遵循以下步骤：

1. 按照需要配置 SPP （包括等待状态和时钟）。
2. 将 SPPOWN 位设置为受 USB 外设控制。
3. 将缓冲器描述符的起始地址（BDnADRL:BDnADRH）设置为 FFFFh。
4. 将 KEN 位（BDnSTAT<5>）置 1，因此缓冲器描述符将被 SIE 永久保持。
5. 将 INCDIS 位（BDnSTAT<4>）置 1，禁止自动递增缓冲器地址。
6. 将 SPPEN 位置 1，使能 SPP 模块。

注：	如果 USB 端点被配置为使用 SPP，此端点的数据传输类型只能是同步传输的。

18.3 单片机控制设置

SPP 也能作为单片机的并行端口。在这种模式下，SPPEPS 寄存器（寄存器 18-3）提供状态和地址写控制。数据被写入或读出 SPPDATA 寄存器。当 SPP 受控于单片机时， SPP 时钟依靠指令时钟（Fosc/4）产生。

要设置 SPP 使之受单片机控制需要遵循以下步骤：

1. 按照需要配置 SPP （包括等待状态和时钟）。
2. 将 SPPOWN 位清零。
3. 将 SPPEN 位置 1，使能 SPP 模块。

18.3.1 SPP 中断

当受控于单片机内核时，SSP 中断控制逻辑会在每次读写操作完成时，产生一个中断来通知应用程序。中断标志位是 SPPIF （PIR1<7>），中断允许位是 SSPIE 位（PIE1<7>）。像单片机的其他电平中断一样，它也可以被设置为具有低或高优先级。这是通过设置 SSPIP 位（IPR1<7>）完成的。

18.3.2 对 SPP 进行写操作

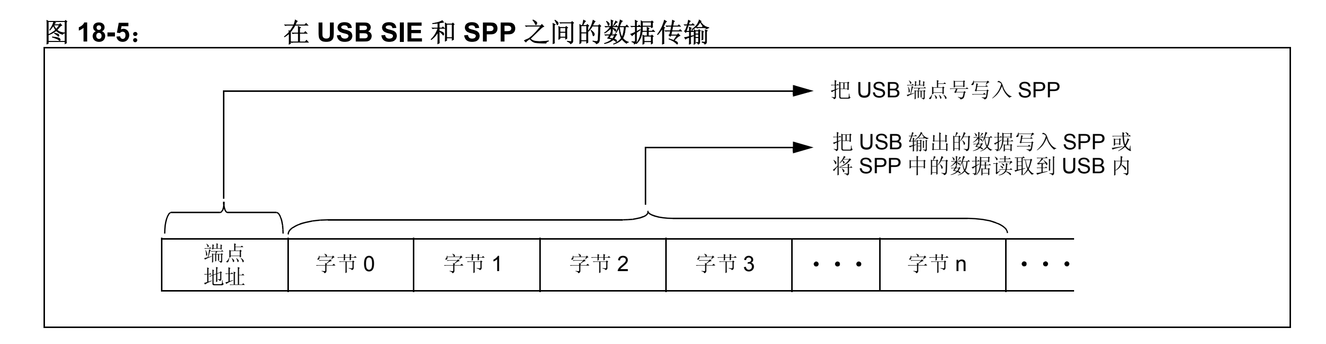
一旦配置完毕，就可以通过对 SPPEPS 和 SPPDATA 寄存器执行写操作来写入 SPP。如果 SPP 被配置为在时钟控制下输出端点地址和数据信息，则对 SPPEPS 寄存器的写操作将启动地址写周期。 另外，通过向 SPPDATA 寄存器写入数据启动数据写周期。SPPBUSY 位表示地址和数据写周期的状态。

下面是写操作序列的一个示例：

1. 向 SPPEPS 寄存器写入 4 位地址。SPP 自动开始地址写入。如果不用写地址，则转到步聚 3。
2. 监视 SSPBUSY 位看地址是否已经发送完毕。持续时间取决于等待状态的个数。
3. 向 SPPDATA 寄存器写入数据。SPP 自动开始写数据。
4. 监视 SSPBUSY 位看数据是否已经发送完毕。持续时间取决于等待状态的个数。
5. 返回到步骤 1 或 3，写入新地址或数据。

注：	应该查询SSPBUSY位，以确保对SPPEPS 或 SPPDATA 寄存器的连续写操作不会超过设定的等待时间。

图 18-5： 在 USB SIE 和 SPP 之间的数据传输



18.3.3 对 SPP 进行读操作

从 SPP 读取数据就是读取 SPPDATA 寄存器中的数据。对该寄存器的首次读取启动读操作。当读操作完成时（由 SSPBUSY 位表示），SPPDATA 将装入当前数据。

下面是读操作序列的一个示例：

1. 向 SPPEPS 寄存器写入 4 位地址。SPP 自动开始地址写入。如果不用写地址，则转到步聚 3。
2. 监视 SSPBUSY 位看地址是否已经发送完毕。持续时间取决于等待状态的个数。
3. 从 SPPDATA 寄存器读取数据，将返回上次读操作读到的数据。SPP 自动启动下一个读周期。
4. 监视 SSPBUSY 位看数据是否已经被读取。持续时间取决于等待状态的个数。
5. 返回到步骤3 从SPP读取当前字节并启动下一个读周期。

寄存器 18-3： SPPEPS：SPP 端点地址和状态寄存器

R-0	R-0	U-0	R-0	R/W-0	R/W-0	R/W-0	R/W-0
RDSPP	WRSPP	—	SPPBUSY	ADDR3	ADDR2	ADDR1	ADDR0
bit 7							bit 0

bit 7 **RDSPP**：SPP 读状态位 （当 SPPCON<SSPOWN> = 1 时有效，受控于 USB）
1 = 上一次发生的事件是读 SPP
0 = 上一次发生的事件不是读 SPP

bit 6 **WRSPP**：SPP 写状态位 （当 SPPCON<SSPOWN> = 1 时有效，受控于 USB）
1 = 上一次发生的事件是写 SPP
0 = 上一次发生的事件不是写 SPP

bit 5 未用：读为 0

bit 4 **SPPBUSY**：SPP 通信状态位
1 = SPP 忙
0 = SPP 正准备接受下一次读 / 写请求

bit 3-0 **ADDR3:ADDR0**：SPP 端点地址位
1111 = 端点地址 15
• •
• •
0001
0000 = 端点地址 0

图注：			
R = 可读位	W = 可写位	U = 未用位，读为 0	
-n = 上电复位时的值	1 = 置 1	0 = 清零	x = 未知

表 18-1： 与并行通信端口相关的寄存器

名称	Bit 7	Bit 6	Bit 5	Bit 4	Bit 3	Bit 2	Bit 1	Bit 0	复位值所在页
SPPCON[3]	—	—	—	—	—	—	SPPOWN	SPPEN	55
SPPCFG[3]	CLKCFG1	CLKCFG0	CSEN	CLK1EN	WS3	WS2	WS1	WS0	55
SPPEPS[3]	RDSPP	WRSPP	—	SPPBUSY	ADDR3	ADDR2	ADDR1	ADDR0	55
SPPDATA[3]	SPP 数据寄存器								55
PIR1	SPPIF[3]	ADIF	RCIF	TXIF	SSPIF	CCP1IF	TMR2IF	TMR1IF	54
PIE1	SPPIE[3]	ADIE	RCIE	TXIE	SSPIE	CCP1IE	TMR2IE	TMR1IE	54
IPR1	SPPIP[3]	ADIP	RCIP	TXIP	SSPIP	CCP1IP	TMR2IP	TMR1IP	54
PORTE	RDPU[3]	—	—	—	RE3[1,2]	RE2[3]	RE1[3]	RE0[3]	54

图注： — = 未用 （读为 0）。并行通信端口不使用阴影单元。

注 1： 仅当主复位功能被禁止 （MCLRE 配置位 = 0）时使用。

2： RE3 是唯一在 28 引脚和 40/44 引脚器件上都存在的 PORTE 位。其他位只有在 40/44 引脚器件上才存在。

3： 28 引脚器件不使用这些寄存器和 / 或位。

19.0 主控同步串口 （MSSP）模块

19.1 主控 SSP （MSSP）模块概述

主控同步串口（Master Synchronous Serial Port，MSSP）模块是用于同其他外设或单片机器件进行通信的串行接口。这些外设包括串行 EEPROM、移位寄存器、显示驱动器和 A/D 转换器等。MSSP 模块可以工作在以下两种模式：

- 串行外设接口 （SPI）
- I^2C
  - 全主控模式
  - 从动模式 （支持广播地址呼叫）

I^2C 接口通过硬件支持下列模式：

- 主控模式
- 多主机模式
- 从动模式

19.2 控制寄存器

MSSP 模块有三个跟它相关的寄存器，包括一个状态寄存器 （SSPSTAT）和两个控制寄存器 （SSPCON1 和 SSPCON2）。取决于 MSSP 模块是在 SPI 模式还是 I^2C 模式下工作，这些寄存器的用途及它们各自的配置位将完全不同。

下面各节提供了更详细的信息。

19.3 SPI 模式

SPI 模式允许同时同步发送和接收 8 位数据。该模块支持 SPI 的所有四种模式。通常使用以下三个引脚来完成通信：

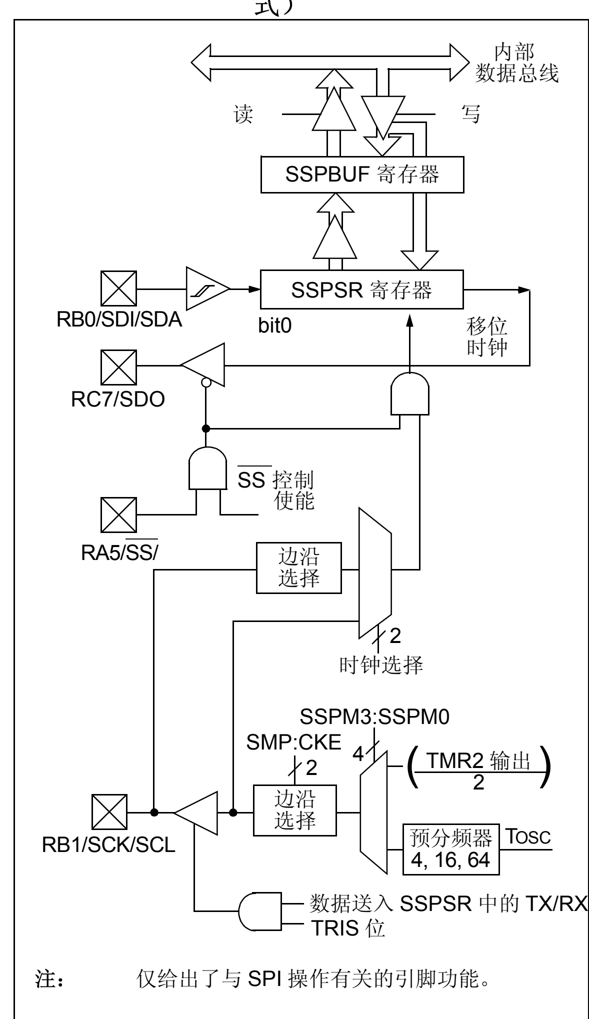
- 串行数据输出 （Serial Data Out， SDO）－ RC7/RX/DT/SDO
- 串行数据输入 （Serial Data In， SDI）－ RB0/AN12/INT0/FLT0/SDI/SDA
- 串行时钟 （Serial Clock， SCK）－ RB1/AN10/INT1/SCK/SCL

此外，当工作在从动模式下时要使用第 4 根引脚：

- 从动选择 （$\overline{SS}$）－ RA5/AN4/$\overline{SS}$/HLVDIN/C2OUT

图 19-1 给出了 MSSP 模块在 SPI 模式下工作的框图。

图 19-1： MSSP 结构框图 （SPI 模式）



注： 仅给出了与 SPI 操作有关的引脚功能。

19.3.1 寄存器

MSSP 模块有四个寄存器用于 SPI 工作模式。这些寄存器包括：

- MSSP 控制寄存器 1 （SSPCON1）
- MSSP 状态寄存器 （SSPSTAT）
- 串行接收 / 发送缓冲寄存器 （SSPBUF）
- MSSP 移位寄存器 （SSPSR）－不能直接访问

SSPCON1 和 SSPSTAT 是在 SPI 模式下工作的控制寄存器和状态寄存器。 SSPCON1 寄存器是可读写的。SSPSTAT 的低 6 位是只读的，其高 2 位是可读写的。

SSPSR 是用来将数据移入或移出的移位寄存器。SSPBUF 是缓冲寄存器，可用于缓存读写的数据字节。

接收时，SSPSR 和 SSPBUF 共同构成一个双重缓冲接收器。当 SSPSR 接收到一个完整的字节后，该字节被送入 SSPBUF 寄存器，同时将中断标志位 SSPIF 置 1。

在发送过程中， SSPBUF 并不是双重缓冲的。对 SSPBUF 的写操作将同时写入 SSPBUF 和 SSPSR。

寄存器 19-1： SSPSTAT：MSSP 状态寄存器 （SPI 模式）

R/W-0	R/W-0	R-0	R-0	R-0	R-0	R-0	R-0
SMP	CKE	D/$\overline{\text{A}}$	P	S	R/$\overline{\text{W}}$	UA	BF
bit 7							bit 0

bit 7 **SMP**：采样位
SPI 主控模式：
1 = 在数据输出时间的末端采样输入数据
0 = 在数据输出时间的中间采样输入数据
SPI 从动模式：
当 SPI 在从动模式下使用时，必须将 SMP 清零。

bit 6 **CKE**：SPI 时钟选择位
1 = 当时钟从有效状态转换到空闲状态时开始发送
0 = 当时钟从空闲状态转换到有效状态时开始发送
时钟状态的极性由 CKP 位 （SSPCON1<4>）设置。

bit 5 **D/$\overline{\text{A}}$**：数据 / $\overline{\text{地址}}$位
仅在 I^2C 模式下使用。

bit 4 **P**：停止位
仅在 I^2C 模式下使用。当禁止 MSSP 模块 （SSPEN 清零）时该位被清零。

bit 3 **S**：起始位
仅在 I^2C 模式下使用。

bit 2 **R/$\overline{\text{W}}$**：读 / $\overline{\text{写}}$信息位
仅在 I^2C 模式下使用。

bit 1 **UA**：更新地址位
仅在 I^2C 模式下使用。

bit 0 **BF**：缓冲器满状态位 （仅用于接收模式）
1 = 接收完成， SSPBUF 满
0 = 接收未完成， SSPBUF 空

图注：			
R = 可读位	W = 可写位	U = 未用位，读为 0	
-n = 上电复位时的值	1 = 置 1	0 = 清零	x = 未知

寄存器 19-2： **SSPCON1：MSSP 控制寄存器 1 （SPI 模式）**

R/W-0	R/W-0	R/W-0	R/W-0	R/W-0	R/W-0	R/W-0	R/W-0
WCOL	SSPOV	SSPEN	CKP	SSPM3	SSPM2	SSPM1	SSPM0
bit 7							bit 0

bit 7 **WCOL**：写冲突检测位 （仅用于发送模式）
1 = 正在发送前一个字时，又有数据写入 SSPBUF 寄存器
（必须由软件清零）
0 = 未发生冲突

bit 6 **SSPOV**：接收溢出指示位
SPI 从动模式：
1 = SSPBUF 中仍保存前一数据时，又接收到一个新的字节。如果溢出，SSPSR 中的数据会丢失。溢出只发生在从动模式下。即使只是发送数据，也必须读 SSPBUF，以避免溢出标志位置 1 （必须由软件清零）。
0 = 无溢出

注： 在主控模式下，溢出位不会被置1，因为每次接收和发送新数据都是通过写SSPBUF寄存器开始的。

bit 5 **SSPEN**：同步串口使能位
1 = 使能串口并将 SCK、 SDO、 SDI 和 $\overline{SS}$ 配置为串口引脚
0 = 禁止串口并将上述引脚配置为 I/O 端口引脚

注： 当该位使能时，必须将这些引脚正确地配置为输入或输出。

bit 4 **CKP**：时钟极性选择位
1 = 空闲状态时，时钟为高电平
0 = 空闲状态时，时钟为低电平

bit 3-0 **SSPM3:SSPM0**：同步串口模式选择位
0101 = SPI 从动模式，时钟 = SCK 引脚，禁止 $\overline{SS}$ 引脚控制功能，可将 $\overline{SS}$ 用作 I/O 引脚
0100 = SPI 从动模式，时钟 = SCK 引脚，使能 $\overline{SS}$ 引脚控制功能
0011 = SPI 主控模式，时钟 = TMR2 输出 /2
0010 = SPI 主控模式，时钟 = Fosc/64
0001 = SPI 主控模式，时钟 = Fosc/16
0000 = SPI 主控模式，时钟 = Fosc/4

注： 在此未列出的位组合用于保留或仅在 I^2C 模式下使用。

图注：			
R = 可读位	W = 可写位	U = 未用位，读为 0	
-n = 上电复位时的值	1 = 置 1	0 = 清零	x = 未知

19.3.2 工作原理

当初始化 SPI 时，需要通过对相应的控制位（SSPCON1<5:0> 和 SSPSTAT<7:6>）编程来设置选项。这些控制位允许设置以下选项：

- 主控模式 （SCK 作为时钟输出）
- 从动模式 （SCK 作为时钟输入）
- 时钟极性 （SCK 处于空闲状态）
- 输入数据的采样相位 （数据输出时间的中间或末端）
- 时钟边沿 （在 SCK 的上升沿 / 下降沿输出数据）
- 时钟速率 （仅主控模式下）
- 从动选择模式 （仅从动模式下）

MSSP 模块由一个发送 / 接收移位寄存器（SSPSR）和一个缓冲寄存器 （SSPBUF）组成。 SSPSR 将数据移入 / 出器件，最高有效位在前。在新数据接收完毕前，SSPBUF 保存上次写入 SSPSR 的数据。一旦 8 位数据接收完毕，该字节就被移入 SSPBUF 寄存器。然后，缓冲器满检测位BF（SSPSTAT<0>）和中断标志位SSPIF 均被置 1。这种双重缓冲数据接收方式 （SSPBUF）允许在读取刚接收的数据之前，就开始接收下一个字节。在数据发送 / 接收期间，任何试图写 SSPBUF 寄存器的操作均无效，并会将写冲突检测位 WCOL （SSPCON1<7>）置 1。用户必须用软件将 WCOL 位清零才能判断后续对 SSPBUF 寄存器的写入是否成功。

为确保应用软件能有效地接收数据，应该在要发送的下一数据字节写入 SSPBUF 之前，读取 SSPBUF 中现有的数据。缓冲器满标志位 BF （SSPSTAT<0>）用于表示何时 SSPBUF 载入了已接收的数据（发送完成）。当 SSPBUF 中的数据被读取后，BF 位即被清零。如果 SPI 仅仅作为一个发送器，则不必理会该位。通常可用 MSSP 中断来判断发送或接收何时完成。必须读取和 / 或写入 SSPBUF。如果不使用中断来处理数据的收发，用软件查询方法同样可确保不会发生写冲突。例 19-1 显示装载 SSPBUF （SSPSR）进行数据发送的过程。

用户不能直接读写 SSPSR 寄存器，只能通过访问 SSPBUF 寄存器来间接访问。此外，MSSP 状态寄存器（SSPSTAT）指示各种状态。

例 19-1： 装载 SSPBUF （SSPSR）寄存器

```
LOOP    BTFSS   SSPSTAT, BF     ;Has data been received (transmit complete)?
        BRA     LOOP            ;No
        MOVF    SSPBUF, W       ;WREG reg = contents of SSPBUF
        MOVWF   RXDATA          ;Save in user RAM, if data is meaningful
        MOVF    TXDATA, W       ;W reg = contents of TXDATA
        MOVWF   SSPBUF          ;New data to xmit
```


19.3.3 使能 SPI I/O

要使能串口，必须将SSP使能位SSPEN（SSPCON1<5>）置 1。要复位或重新配置 SPI 模式，要先将 SSPEN 位清零，重新初始化 SSPCON 寄存器，然后将 SSPEN 位置 1。这将把 SDI、 SDO、 SCK 和 $\overline{SS}$ 引脚配置为串口引脚。要让上述引脚充当串口引脚，必须按如下说明正确设置其中一些引脚的数据方向位（在 TRIS 寄存器中）：

- SDI 由 SPI 模块自动控制
- SDO 必须将 TRISC<7> 位清零
- SCK （主控模式）必须将 TRISB<1> 位清零
- SCK （从动模式）必须将 TRISB<1> 位置 1
- $\overline{SS}$ 必须将 TRISA<5> 位置 1

对于不需要的任何串口功能，可通过将对应的数据方向（TRIS）寄存器设置为相反值来屏蔽。

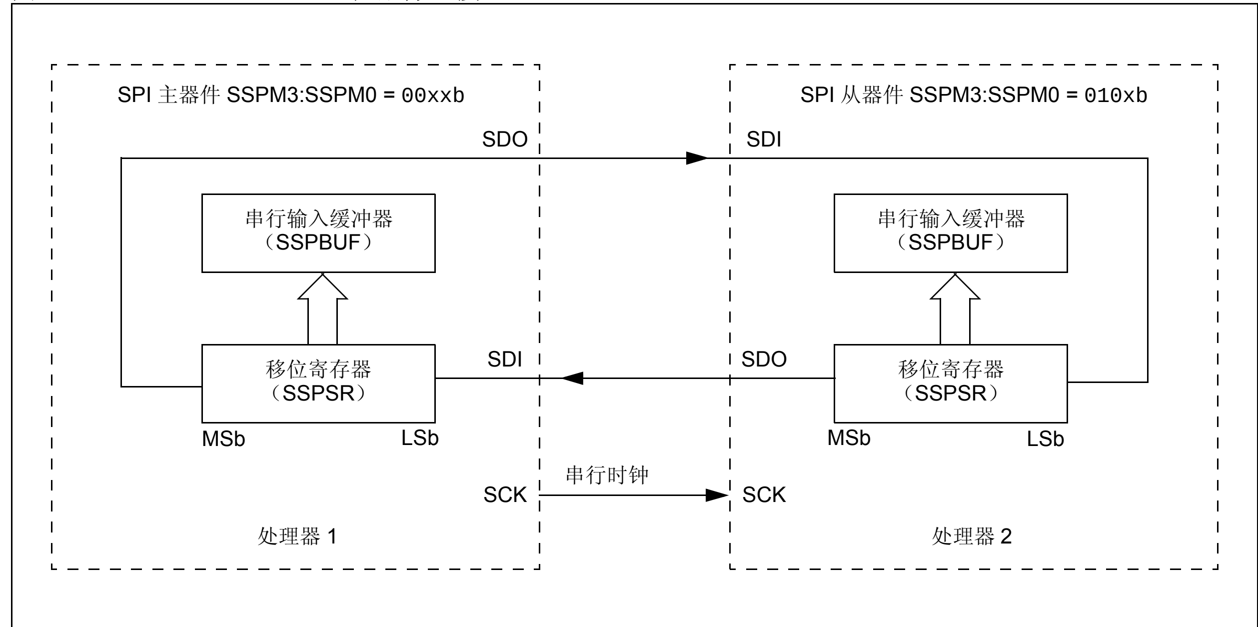
19.3.4 典型连接

图 19-2 给出了两个单片机之间的典型连接。主控制器（处理器 1）通过发送 SCK 信号来启动数据传输。数据在编程设定的时钟边沿被传送，并在相反的时钟边沿被锁存。必须将两个处理器的时钟极性（CKP）设定为相同，这样就可以同时收发数据。数据是否有效取决于应用软件。这就导致以下三种数据传输情形：

- 主器件发送数据 — 从器件发送无效 （Dummy）数据
- 主器件发送数据 — 从器件发送数据
- 主器件发送无效数据 — 从器件发送数据



图 19-2： SPI 主 / 从器件连接



19.3.5 主控模式

主器件控制 SCK 信号，所以可以随时启动数据传输。主器件根据软件协议确定从器件（处理器 2，图 19-2）应在何时广播数据。

在主控模式下，数据一旦写入 SSPBUF 寄存器就开始发送或接收数据。如果只打算将 SPI 作为接收器，则可以禁止 SDO 输出（通过编程将其设置为输入）。SSPSR 寄存器按设定的时钟速率连续移入 SDI 引脚上的信号。每收到一个字节，就将其装入 SSPBUF 寄存器，就像接收到普通字节一样 （中断和状态位相应置 1）。这在以 "线路活动监控"（Line Activity Monitor）方式工作的接收器应用中很有用。

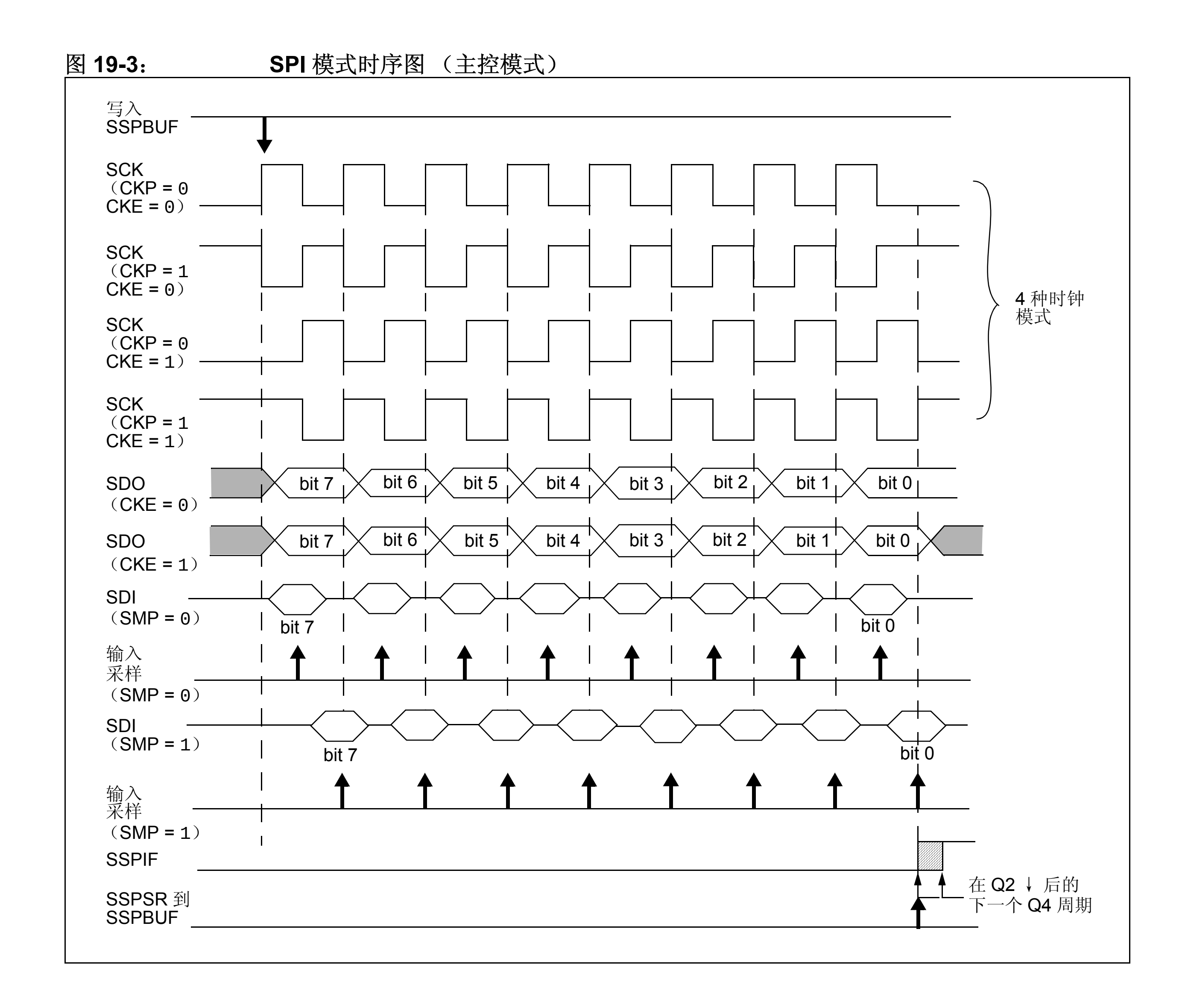
时钟极性可通过对 CKP 位 （SSPCON<4>）进行适当的编程来选择。图 19-3、图 19-5 和图 19-6 将给出 SPI 通信的时序图，其中最先发送最高有效位。在主控模式下， SPI 时钟速率（比特率）可由用户编程设定为下面速率之一：

- FOSC/4 （或 TCY）
- FOSC/16 （或 4 TCY）
- FOSC/64 （或 16 TCY）
- Timer2 输出 /2

这样数据最大传输速率为 10.00 Mbps （时钟速率为 40MHz）。

图 19-3 给出了主控模式的时序图。当 CKE 位置 1 时， SDO 数据在 SCK 出现时钟边沿前一直有效。图中所示输入采样的变化由 SMP 位的状态决定。图中指出了接收到的数据装入 SSPBUF 的时间。

图 19-3： SPI 模式时序图 （主控模式）



 初稿

19.3.6 从动模式

在从动模式下，当 SCK 引脚上有外部时钟脉冲时发送和接收数据。当最后一位数据被锁存后，中断标志位 SSPIF 置 1。

在 SPI 从动模式下使能该模块之前，时钟线必须符合正确的空闲状态。可以通过读 SCK 引脚来观察时钟线。由 CKP 位（SSPCON1<4>）确定空闲状态。

在从动模式下，外部时钟由 SCK 引脚上的外部时钟源提供。外部时钟必须满足电气规范中规定的高电平和低电平的最短时间要求。

在休眠状态下，从器件仍可发送 / 接收数据。当收到一个字节时，将唤醒休眠状态下的器件。

19.3.7 从动选择同步

$\overline{SS}$ 引脚允许模块工作在同步从动模式下。要进入该模式，SPI 必须处于从动模式，并使能 $\overline{SS}$ 引脚控制（SSPCON1<3:0> = 04h）。要让 $\overline{SS}$ 引脚充当输入引脚，则不能将此引脚驱动为低电平。数据锁存器必须为高电平。当 $\overline{SS}$ 引脚为低电平时，使能数据的发送和接收，同时驱动 SDO 引脚。当 $\overline{SS}$ 引脚变为高电平时，即使是在字节的发送过程中，SDO 引脚也不再被驱动，而是变成高阻悬空状态。根据应用需要，可在引脚上外接上拉 / 下拉电阻。

> 注 1：当 SPI 处于从动模式，并且使能 $\overline{SS}$ 引脚控制功能（SSPCON1<3:0> = 0100）时，将 $\overline{SS}$ 引脚置为 V_{DD} 电平，会使 SPI 模块复位。
>
> 2：如果 SPI 工作在从动模式并且 CKE 置 1，则必须使能 $\overline{SS}$ 引脚控制功能。

当 SPI 模块复位后，位计数器被强制为 0。这可以通过强制将 $\overline{SS}$ 引脚拉为高电平或将 SSPEN 位清零来实现。

将 SDO 引脚和 SDI 引脚相连，可以仿真二线制通信。当 SPI 需要作为接收器工作时，SDO 引脚可以被配置为输入端。这样就禁止了从 SDO 发送数据。因为 SDI 不会引起总线冲突，因而总是可以将其保留为输入（SDI 功能）。

图 19-4： 从动同步模式的时序图

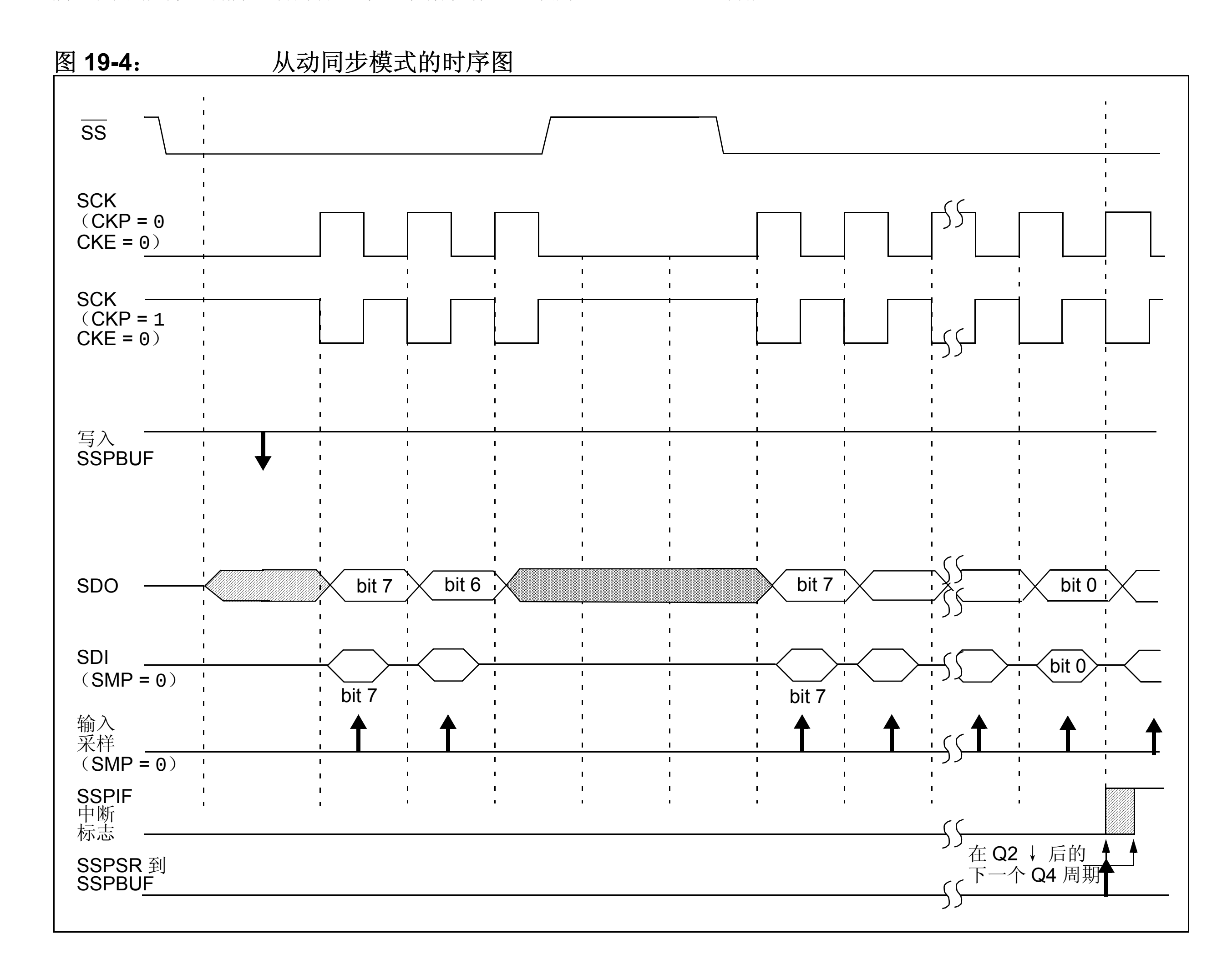


图 19-5： SPI 模式的时序图 （从动模式且 CKE = 0）

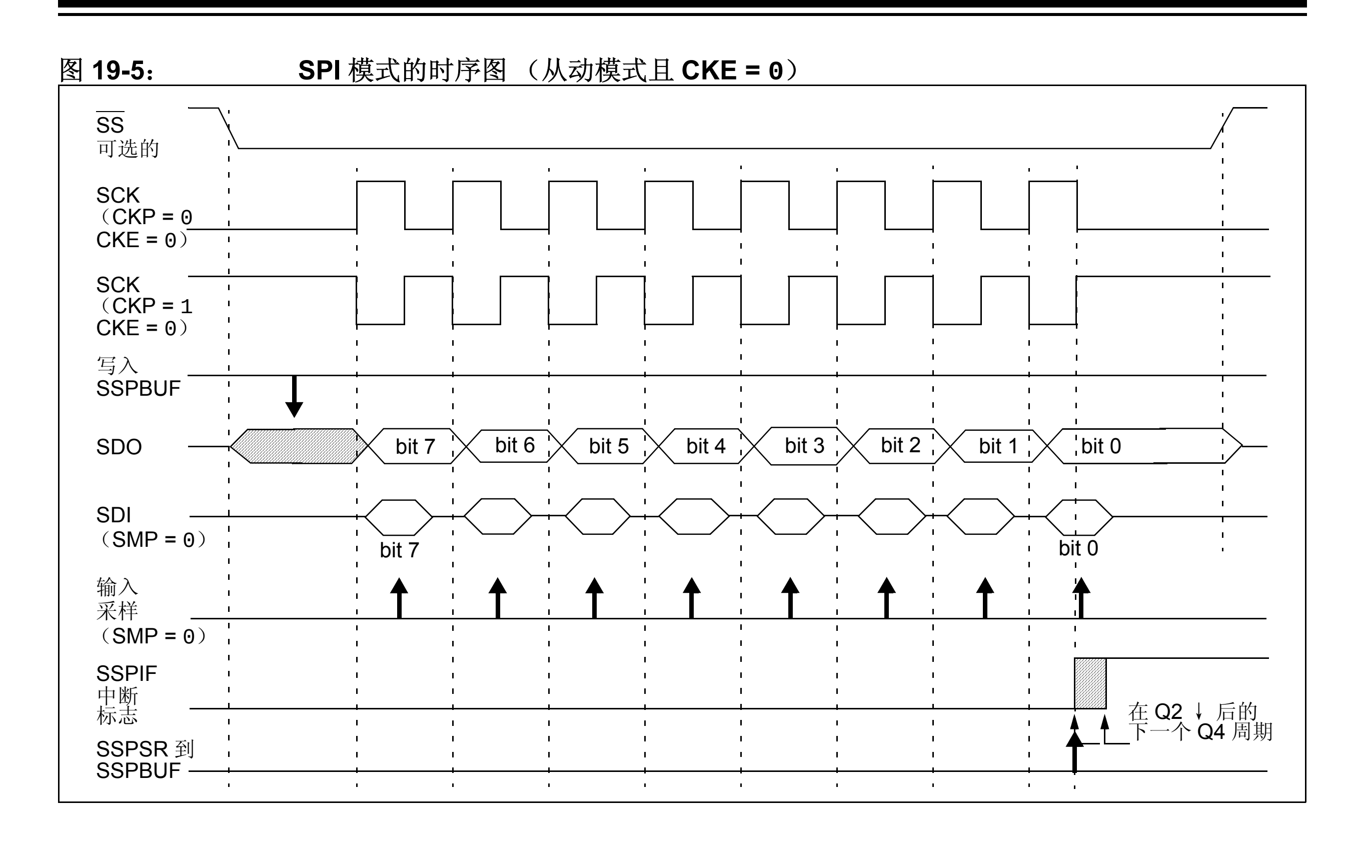
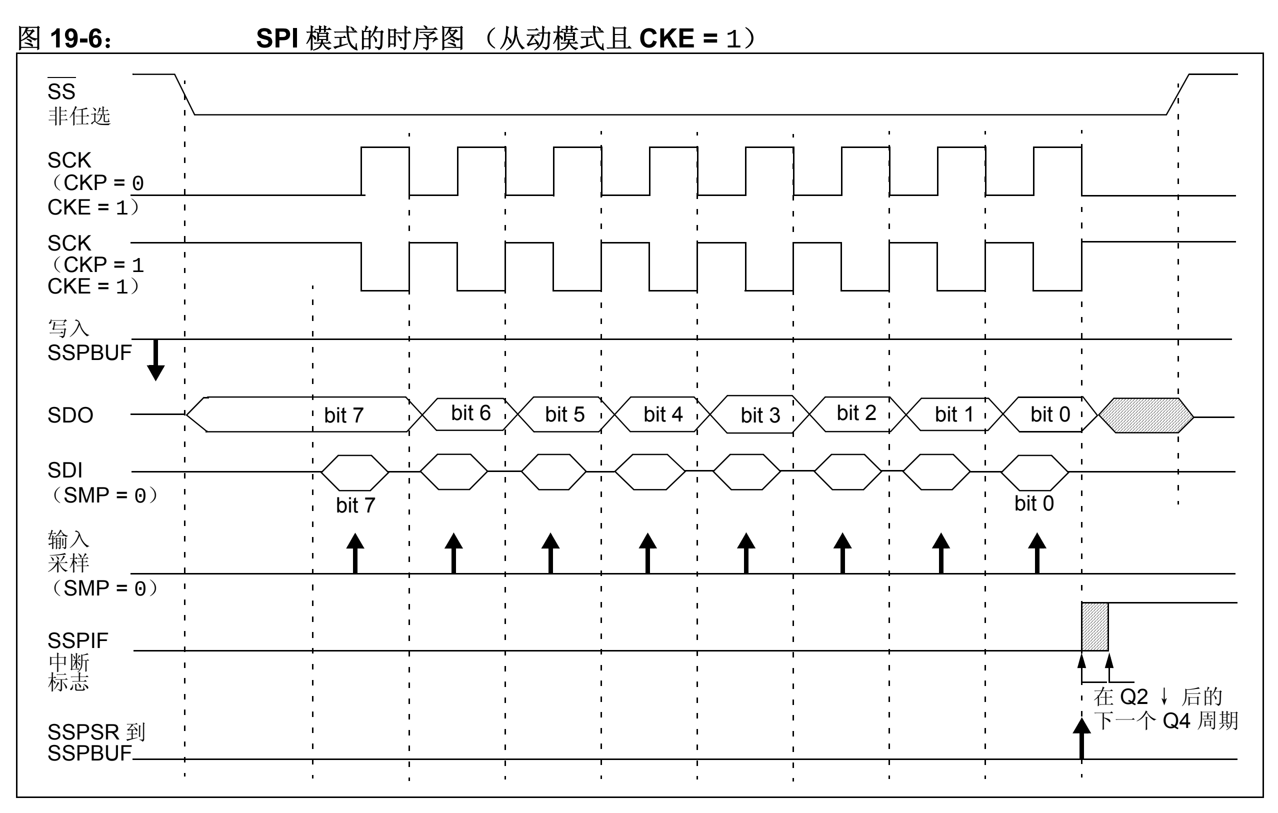


图 19-6： SPI 模式的时序图 （从动模式且 CKE = 1）



 初稿

19.3.8 在功耗管理模式下的工作方式

在 SPI 主控模式下，模块时钟速率与全功耗模式下的速率不同；处于休眠模式时，所有时钟都停止。

在大多数功耗管理模式下需要给外设提供时钟。该时钟可以来自主时钟源、辅助时钟源（32.768kHz 的 Timer1 振荡器）或 INTOSC 时钟源。更多信息请参见**第 2.4 节 "时钟源与振荡器切换"**。

在大多数情况下，主器件为 SPI 数据提供的时钟速率并不重要；但是，每个系统都应该评估此因素。

如果允许了 MSSP 中断，那么当主器件发送完数据时，这些中断可以将控制器从休眠模式或某种空闲模式唤醒。如果不想从休眠或空闲模式退出，应该禁止 MSSP 中断。

如果选择了休眠模式，所有模块的时钟都将停止，并且在器件被唤醒前，发送 / 接收将保持此停滞状态。当器件返回到运行模式后，模块将恢复发送和接收数据。

在 SPI 从动模式下， SPI 发送 / 接收移位寄存器与器件异步工作。这可以使器件处于任何功耗管理模式下，而数据仍可被移入 SPI 发送 / 接收移位寄存器。当 8 位数据全部接收到后，MSSP 中断标志位将置 1 ；如果允许中断的话，将唤醒器件。

19.3.9 复位的影响

复位会禁止 MSSP 模块并终止当前的传输。

19.3.10 总线模式兼容性

表 19-1 中所示是标准 SPI 模式与 CKP 和 CKE 控制位状态的对应关系。

表 19-1： SPI 总线模式

标准 SPI 模式术语	控制位状态	
	CKP	CKE
0, 0	0	1
0, 1	0	0
1, 0	1	1
1, 1	1	0

还有一个 SMP 位用来控制何时采样数据。

表 19-2： 和 SPI 操作相关的寄存器

名称	Bit 7	Bit 6	Bit 5	Bit 4	Bit 3	Bit 2	Bit 1	Bit 0	复位值所在页
INTCON	GIE/GIEH	PEIE/GIEL	TMR0IE	INT0IE	RBIE	TMR0IF	INT0IF	RBIF	51
PIR1	SPPIF(1)	ADIF	RCIF	TXIF	SSPIF	CCP1IF	TMR2IF	TMR1IF	54
PIE1	SPPIE(1)	ADIE	RCIE	TXIE	SSPIE	CCP1IE	TMR2IE	TMR1IE	54
IPR1	SPPIP(1)	ADIP	RCIP	TXIP	SSPIP	CCP1IP	TMR2IP	TMR1IP	54
TRISA	—	TRISA6(2)	TRISA5	TRISA4	TRISA3	TRISA2	TRISA1	TRISA0	54
TRISB	TRISB7	TRISB6	TRISB5	TRISB4	TRISB3	TRISB2	TRISB1	TRISB0	54
TRISC	TRISC7	TRISC6	—	—	—	TRISC2	TRISC1	TRISC0	54
SSPBUF	同步串口接收缓冲器 / 发送寄存器								52
SSPCON1	WCOL	SSPOV	SSPEN	CKP	SSPM3	SSPM2	SSPM1	SSPM0	52
SSPSTAT	SMP	CKE	D/$\overline{A}$	P	S	R/$\overline{W}$	UA	BF	52

图注： — = 未用位 （读为 0）。在 SPI 模式下 MSSP 不使用阴影单元。

注 1： 这些位在 28 引脚器件上未用，应始终保持这些位清零。

2： 根据不同的主振荡模式， RA6 被配置为端口引脚。当禁止端口引脚时，所有相关位均读为 0。

19.4 I2C 模式

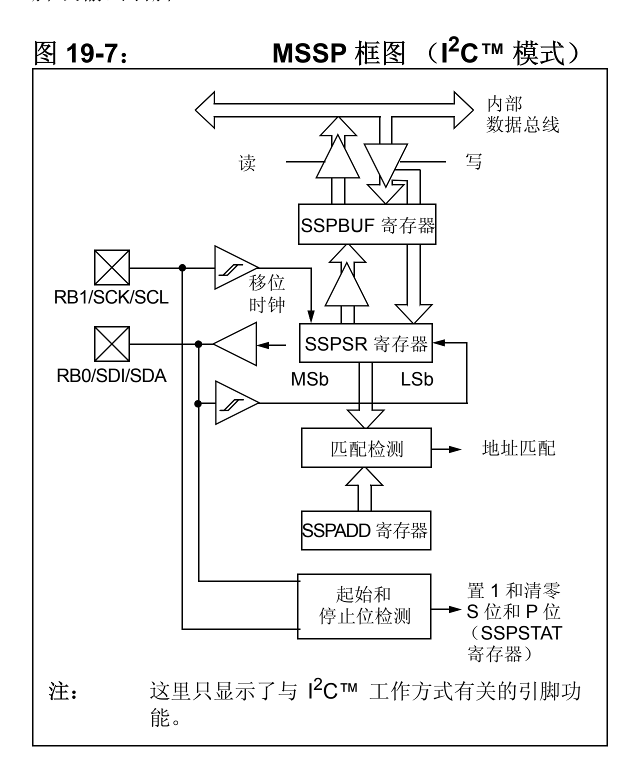
MSSP 模块工作在 I2C 模式时，可以实现所有的主控和从动功能 （包括广播呼叫支持），并且在硬件上提供起始位和停止位的中断用于判断总线何时空闲（多主机功能）。MSSP 模块实现了标准模式规范以及 7 位和 10 位寻址。

有两个引脚用于数据传输：

- 串行时钟 （SCL） － RB1/AN10/INT1/SCK/SCL
- 串行数据 （SDA） － RB0/AN12/INT0/FLT0/SDI/SDA

用户必须通过 TRISB<1:0> 位将这些引脚配置为输入引脚或输出引脚。

图 19-7： MSSP 框图 （I2C™ 模式）



注： 这里只显示了与 I2C™ 工作方式有关的引脚功能。

19.4.1 寄存器

MSSP 模块有 6 个寄存器用于 I2C 工作模式。这些寄存器包括：

- MSSP 控制寄存器 1 （SSPCON1）
- MSSP 控制寄存器 2 （SSPCON2）
- MSSP 状态寄存器 （SSPSTAT）
- 串行接收 / 发送缓冲寄存器 （SSPBUF）
- MSSP 移位寄存器 （SSPSR）－不能直接访问
- MSSP 地址寄存器 （SSPADD）

SSPCON1、SSPCON2 和 SSPSTAT 是在 I2C 模式下工作的控制寄存器和状态寄存器。SSPCON1 和 SSPCON2 寄存器是可读写的。SSPSTAT 的低 6 位是只读的，其高 2 位是可读写的。

SSPSR 是用来将数据移入或移出的移位寄存器。SSPBUF 是缓冲寄存器，可用于缓存读写的数据字节。

在 I2C 从动模式下配置 SSP 时， SSPADD 寄存器将保存从器件的地址。在主控模式下配置 SSP 时，SSPADD 的低 7 位保存波特率发生器的重载值。

接收时，SSPSR 和 SSPBUF 共同构成一个双重缓冲接收器。当 SSPSR 接收到一个完整的字节后，该字节被送入 SSPBUF 寄存器，同时将中断标志位 SSPIF 置 1。

在发送过程中， SSPBUF 并不是双重缓冲的。对 SSPBUF 的写操作将同时写入 SSPBUF 和 SSPSR。

 初稿

寄存器 19-3： **SSPSTAT：MSSP 状态寄存器 （I²C™ 模式）**

R/W-0	R/W-0	R-0	R-0	R-0	R-0	R-0	R-0
SMP	CKE	D/$\overline{A}$	P	S	R/$\overline{W}$	UA	BF
bit 7							bit 0

bit 7 **SMP：**变化率控制位
在主控或从动模式下：
1 = 标准速率模式下禁止变化率控制 （100 kHz 和 1 MHz）
0 = 高速模式下使能变化率控制 （400 kHz）

bit 6 **CKE：**SMBus 选择位
在主控或从动模式下：
1 = 使能 SMBus 特定输入
0 = 禁止 SMBus 特定输入

bit 5 **D/$\overline{A}$：**数据 / $\overline{地址}$位
在主控模式下：
保留。
在从动模式下：
1 = 表示上一个接收或发送的字节是数据
0 = 表示上一个接收或发送的字节是地址

bit 4 **P：**停止位
1 = 表示上一次检测到了停止位
0 = 表示上一次未检测到停止位

注： 当复位或 SSPEN 被清零时该位清零。

bit 3 **S：**起始位
1 = 表示上一次检测到了起始位
0 = 表示上一次未检测到起始位

注： 当复位或 SSPEN 被清零时该位清零。

bit 2 **R/$\overline{W}$：**读 / $\overline{写}$信息位 （仅用于 I²C 模式）
在从动模式下：
1 = 读
0 = 写

注： 该位用来保存在上一次地址匹配后的 R/$\overline{W}$ 位信息。该位仅在从地址匹配开始到下一个起始位、停止位或非 ACK 位之间有效。

在主控模式下：
1 = 正在进行发送
0 = 未进行发送

注： 该位与 SEN、RSEN、PEN、RCEN 或 ACKEN 逻辑或运算的结果表示 MSSP 是否处于空闲模式。

bit 1 **UA：**更新地址位 （仅用于 10 位从动模式）
1 = 表示用户需要更新 SSPADD 寄存器中的地址
0 = 不需要更新地址

bit 0 **BF：**缓冲器满状态位
在发送模式下：
1 = 接收完成， SSPBUF 满
0 = 接收未完成， SSPBUF 空
在接收模式下：
1 = 正在发送数据 （不包括 $\overline{ACK}$ 和停止位）， SSPBUF 已满
0 = 数据发送完成 （不包括 ACK 位和停止位）， SSPBUF 为空

图注：			
R = 可读位	W = 可写位	U = 未用位，读为 0	
-n = 上电复位时的值	1 = 置 1	0 = 清零	x = 未知

寄存器 19-4： SSPCON1：MSSP 控制寄存器 1 （I^2C 模式）

R/W-0	R/W-0	R/W-0	R/W-0	R/W-0	R/W-0	R/W-0	R/W-0
WCOL	SSPOV	SSPEN	CKP	SSPM3	SSPM2	SSPM1	SSPM0
bit 7							bit 0

bit 7 **WCOL**：写冲突检测位

在主控发送模式下：

1 = 当 I^2C 不满足启动传输条件时，尝试写 SSPBUF 寄存器 （必须由软件清零）

0 = 未发生冲突

在从动发送模式下：

1 = 正在发送前一个字时，又有数据写入 SSPBUF 寄存器 （必须由软件清零）

0 = 未发生冲突

在接收模式下 （主控或从动模式）：

此位被忽略。

bit 6 **SSPOV**：接收溢出指示位

在接收模式下：

1 = SSPBUF 寄存器仍在保存前一字节时，又接收到一个新的字节 （必须由软件清零）

0 = 未发生溢出

在发送模式下：

在发送模式下，此位被忽略。

bit 5 **SSPEN**：同步串口使能位

1 = 使能串口并将 SDA 和 SCL 引脚配置为串口引脚

0 = 禁止串口并将这些引脚配置为 I/O 端口引脚

注： 当使能该位时，必须将 SDA 和 SCL 引脚正确地配置为输入引脚或输出引脚。

bit 4 **CKP**：SCK 释放控制位

在从动模式下：

1 = 释放时钟

0 = 保持时钟低电平 （时钟延长），用来确保数据建立时间

在主控模式下：

在此模式下未使用。

bit 3-0 **SSPM3:SSPM0**：同步串口模式选择位

1111 = I^2C 从动模式， 10 位地址，并使能起始位和停止位中断

1110 = I^2C 从动模式， 7 位地址，并使能起始位和停止位中断

1011 = I^2C 由固件控制的主控模式 （从器件空闲）

1000 = I^2C 主控模式，时钟 = Fosc/(4 * (SSPADD + 1))

0111 = I^2C 从动模式， 10 位地址

0110 = I^2C 从动模式， 7 位地址

注： 此处未列出的组合为保留或只用于 SPI 模式。

图注：			
R = 可读位	W = 可写位	U = 未用位，读为 0	
-n = 上电复位时的值	1 = 置 1	0 = 清零	x = 未知

寄存器 19-5： SSPCON2：MSSP 控制寄存器 2 （I^2C 模式）

R/W-0	R/W-0	R/W-0	R/W-0	R/W-0	R/W-0	R/W-0	R/W-0
GCEN	ACKSTAT	ACKDT	ACKEN(1)	RCEN(1)	PEN(1)	RSEN(1)	SEN(1)
bit 7							bit 0

bit 7 **GCEN**：广播呼叫使能位 （仅用于从动模式）
1 = 当 SSPSR 接收到广播呼叫地址 （0000h）时允许中断
0 = 禁止广播呼叫地址

bit 6 **ACKSTAT**：应答状态位 （仅用于主控发送模式）
1 = 未收到来自从器件的应答
0 = 收到来自从器件的应答

bit 5 **ACKDT**：应答数据位 （仅用于主控接收模式）
1 = 无应答
0 = 应答

注： 用户在接收结束时发起一个应答序列，同时发送该值。

bit 4 **ACKEN**：应答序列使能位 （仅主控接收模式）(1)
1 = 在 SDA 和 SCL 引脚上发起应答序列，并发送 ACKDT 数据位。由硬件自动清零。
0 = 应答序列空闲

bit 3 **RCEN**：接收使能位 （仅用于主控模式）(1)
1 = 使能 I^2C 接收模式
0 = 接收空闲

bit 2 **PEN**：停止条件使能位 （仅用于主控模式）(1)
1 = 在 SDA 和 SCL 引脚上发起停止条件。由硬件自动清零。
0 = 停止条件空闲

bit 1 **RSEN**：重复启动条件使能位 （仅用于主控模式）(1)
1 = 在 SDA 和 SCL 引脚发起重复启动条件。由硬件自动清零。
0 = 重复启动条件空闲

bit 0 **SEN**：启动条件使能 / 延长使能位 (1)
在主控模式下：
1 = 在 SDA 和 SCL 引脚上发起启动条件。由硬件自动清零。
0 = 启动条件空闲
在从动模式下：
1 = 为从动发送和从动接收使能时钟延长
0 = 时钟延长被禁止

注 **1**：对于 ACKEN、RCEN、PEN、RSEN 和 SEN 位来说，如果 I^2C 模块不处于空闲模式，这些位就不能被置 1 （不支持后台操作），并且不能对 SSPBUF 进行写操作（禁止写 SSPBUF）。

图注：			
R = 可读位	W = 可写位	U = 未用位，读为 0	
-n = 上电复位时的值	1 = 置 1	0 = 清零	x = 未知

19.4.2 工作原理

通过将 MSSP 使能位 SSPEN（SSPCON1<5>）置 1，可使能 MSSP 模块功能。

SSPCON1 寄存器用于控制 I^{2}C 的工作模式。可通过设置 4 个模式选择位 （SSPCON1<3:0>）选择以下一种 I^{2}C 模式：

- I^{2}C 主控模式
- I^{2}C 从动模式 （7 位地址）
- I^{2}C 从动模式 （10 位地址）
- I^{2}C 从动模式 （7 位地址），允许起始位和停止位中断
- I^{2}C 从动模式 （10 位地址），允许起始位和停止位中断
- I^{2}C 由固件控制的主控模式，从器件空闲

在 SSPEN 位置 1 时选择任何 I^{2}C 模式，将强制 SCL 和 SDA 引脚漏极开路；通过将相应的 TRISB 位置 1，将 SCL 和 SDA 引脚编程为输入引脚。要确保此模块正常工作，必须为 SCL 和 SDA 引脚提供外接上拉电阻。

19.4.3 从动模式

在从动模式下， SCL 引脚和 SDA 引脚必须配置为输入（TRISB<1:0> 置 1）。必要时（从发送器）MSSP 模块将使用输出数据改写输入状态。

I^{2}C 从动模式硬件总是在地址匹配时产生中断。用户也可以通过模式选择位选择起始位或停止位中断。

当地址匹配时或在地址匹配后发送的数据被接收时，硬件会自动产生一个应答（$\overline{ACK}$）脉冲，并把当前 SSPSR 寄存器中接收到的值装入 SSPBUF 寄存器。

只要满足下列条件之一， MSSP 模块就不会产生 $\overline{ACK}$ 脉冲：

- 缓冲器满标志位 BF （SSPSTAT<0>）在接收到传输的数据前置 1。
- 溢出标志位 SSPOV （SSPCON1<6>）在接收到传输的数据前置 1。

在这种情况下，SSPSR寄存器的值不会载入SSPBUF，但是 SSPIF 位（PIR1<3>）仍会置 1。BF 位是通过读取 SSPBUF 寄存器清零的，而 SSPOV 位是通过软件清零的。

为确保正常工作，SCL 时钟输入必须满足最小高电平和最小低电平时间要求。时序参数 100 和参数 101 给出了 I^{2}C 规范所规定的高电平和低电平时间以及对 MSSP 模块的具体要求 （见表 28-20）。

19.4.3.1 寻址

一旦使能了 MSSP 模块，它就会等待启动条件出现。启动条件出现后，就会向 SSPSR 寄存器移入 8 位数据。在时钟信号（SCL）的上升沿采样所有的输入位。在第 8 个时钟（SCL）脉冲的下降沿，寄存器 SSPSR<7:1> 的值会和 SSPADD 地址寄存器的值进行比较。如果地址匹配，并且 BF 位和 SSPOV 位都被清零，会发生下列事件 ：

1. SSPSR 寄存器的值被装入 SSPBUF 寄存器。
2. 缓冲器满标志位 BF 置 1。
3. 产生 $\overline{ACK}$ 脉冲。
4. 在第 9 个 SCL 脉冲的下降沿， MSSP 中断标志位 SSPIF （PIR1<3>）被置 1 （如果允许中断，则产生中断）。

在 10 位地址模式下，从器件需要接收两个地址字节。第一个地址字节的高 5 位指定这是否为 10 位地址。R/$\overline{W}$ 位（SSPSTAT<2>）必须指定写操作，这样从器件才能接收到第二个地址字节。对于 10 位地址，第一个字节应该是 `11110 A9 A8 0`，其中 A9 和 A8 是 10 位地址的高 2 位。 10 位地址传输的操作步骤如下，其中 7-9 步是针对从动发送器而言的：

1. 接收地址的第一个 （高）字节 （SSPIF 位、BF 位和 UA 位 （SSPSTAT<1>）置 1）。
2. 用地址的第二个 （低）字节更新 SSPADD 寄存器 （UA 位清零并释放 SCL 线）。
3. 读 SSPBUF 寄存器 （BF 位清零）并将标志位 SSPIF 清零。
4. 接收地址的第二个 （低）字节 （SSPIF 位、BF 位和 UA 位置 1）。
5. 使用地址的第一个 （高）字节更新 SSPADD 寄存器。如果匹配的话就释放 SCL 线，这将清零 UA 位。
6. 读 SSPBUF 寄存器 （BF 位清零）并将标志位 SSPIF 清零。
7. 接收重复启动条件。
8. 接收地址的第一个 （高）字节 （SSPIF 位和 BF 位置 1）。
9. 读 SSPBUF 寄存器 （BF 位清零）并将标志位 SSPIF 清零。

19.4.3.2 接收

当地址字节的 R/$\overline{W}$ 位清零并发生地址匹配时，SSPSTAT 寄存器的 R/$\overline{W}$ 位清零。接收到的地址被装入 SSPBUF 寄存器，且 SDA 线保持低电平 （$\overline{ACK}$）。

当存在地址字节溢出条件时，则会发出无应答脉冲（No ACK）。溢出条件是指 BF 位 （SSPSTAT<0>）置 1，或者 SSPOV 位 （SSPCON1<6>）置 1。

每个数据传输字节都会产生一个 MSSP 中断。标志位 SSPIF （PIR1<3>）必须由软件清零。使用 SSPSTAT 寄存器可以确定该字节的状态。

如果 SEN 被使能 （SSPCON2<0> = 1）， RB1/AN10/INT1/SCK/SCL 将在每个数据传输之后保持为低电平（时钟延长）。必须通过将 CKP 位 （SSPCON1<4>）置 1 才能释放时钟。更多信息请参见第 **19.4.4** 节 **“时钟延长”**。

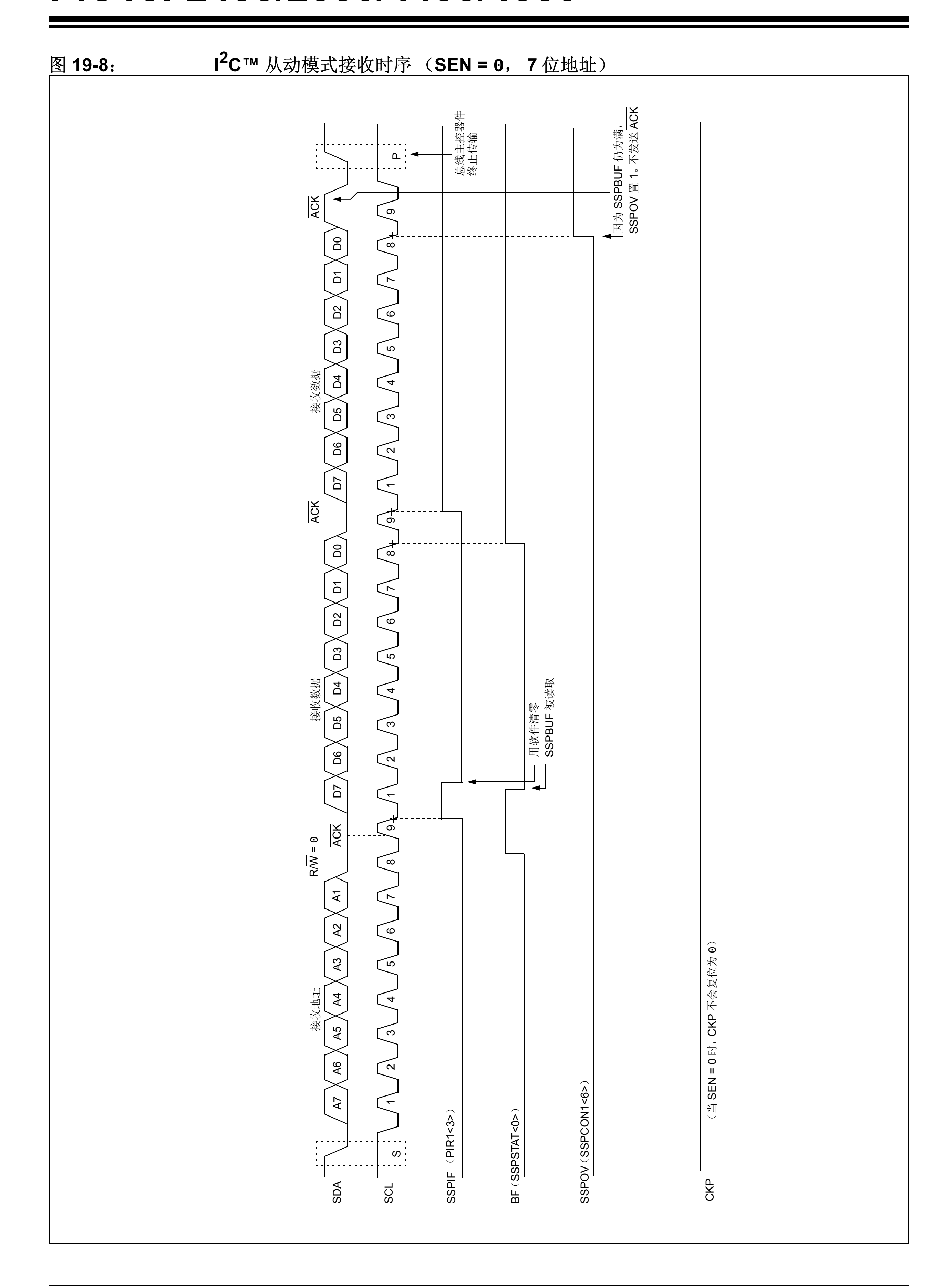
19.4.3.3 发送

当接收到的地址字节的 R/$\overline{W}$ 位置 1 并发生地址匹配时，SSPSTAT 寄存器的 R/$\overline{W}$ 位置 1。接收到的地址被装入 SSPBUF 寄存器。 $\overline{ACK}$ 脉冲在第 9 位上发送，同时不论 SEN 为何值， RB1/AN10/INT1/SCK/SCL 引脚都将保持低电平 （如需了解更多详情，请参见第 **19.4.4** 节 **“时钟延长”**）。通过时钟延长，主器件只有在从器件准备好发送数据时才能发送另一个时钟脉冲。发送的数据必须被装入 SSPBUF 寄存器，同时也被装入了 SSPSR 寄存器。然后，应该通过将 CKP （SSPCON1<4>）置 1 来使能 RB1/AN10/INT1/SCK/SCL 引脚。 8 个数据位会在 SCL 时钟输入的下降沿被移出。这可确保在 SCL 为高电平期间 SDA 信号是有效的 （图 19-9）。

来自主控接收器的 $\overline{ACK}$ 脉冲将在第 9 个 SCL 输入脉冲的上升沿被锁存。如果 SDA 数据信号线为高电平 （无 ACK），那么表示数据传输已完成。在这种情况下，如果从器件锁存了 ACK，将复位从动逻辑 （复位 SSPSTAT 寄存器），同时从器件监视下一个起始位的出现。如果 SDA 线为低电平（$\overline{ACK}$），则必须将下一个要发送的数据装入 SSPBUF 寄存器。然后，通过将 CKP （SSPCON1<4>）置 1 来使能 RB1/AN10/INT1/SCK/SCL 引脚。

每个数据传输字节都会产生一个 MSSP 中断。SSPIF 位必须用软件清零， SSPSTAT 寄存器用于确定字节的状态。 SSPIF 位在第 9 个时钟脉冲的下降沿被置 1。

图 19-8： I²C™ 从动模式接收时序 （SEN = 0， 7 位地址）



 初稿

图 19-9： I^2C™ 从动模式发送时序 （7 位地址）

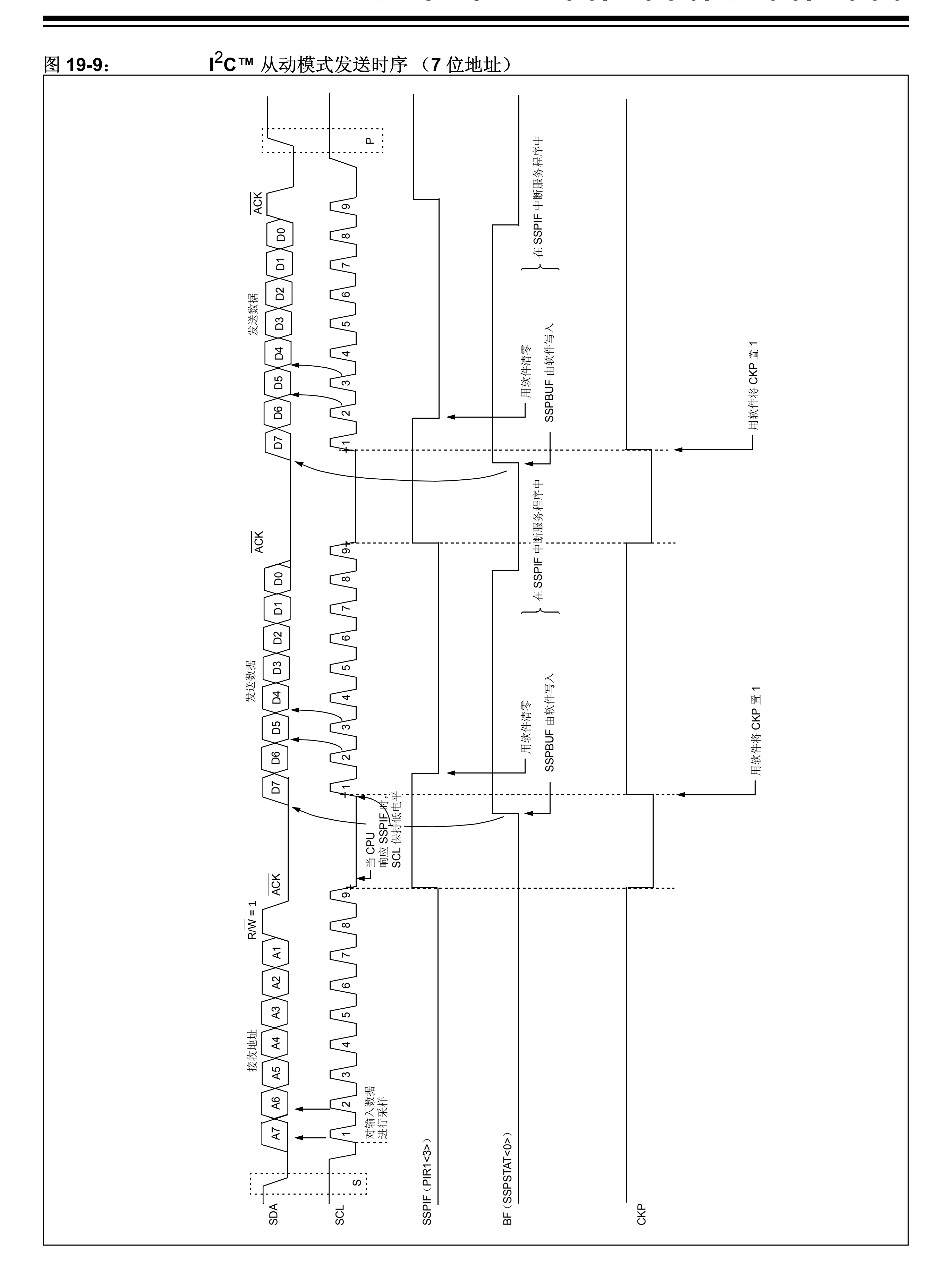
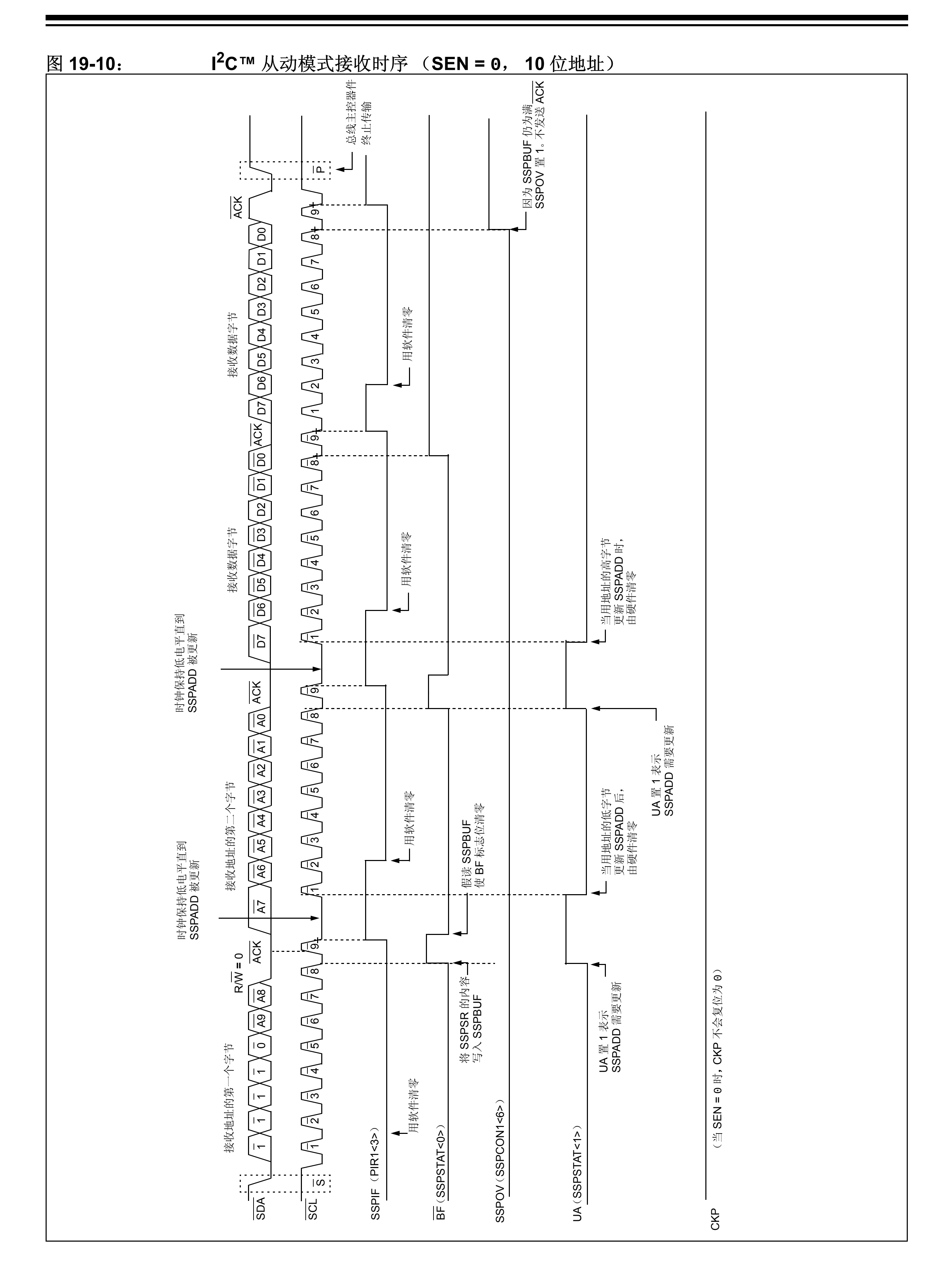
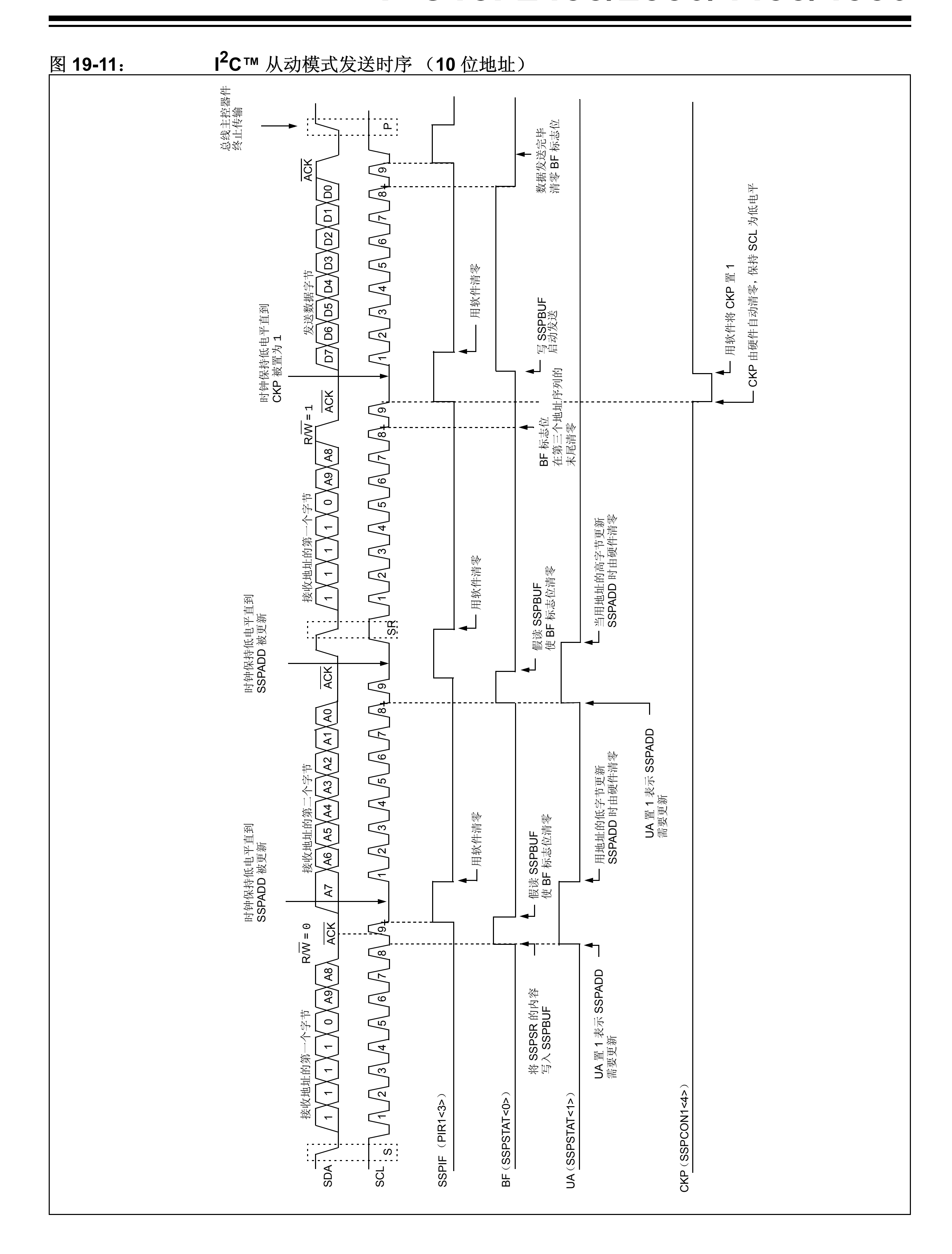


图 19-10： I²C™ 从动模式接收时序 （SEN = 0， 10 位地址）



 初稿

图 19-11： I²C™ 从动模式发送时序 （10 位地址）



19.4.4 时钟延长

7位和10位从动模式都在发送序列中实现了自动时钟延长。

SEN 位 （SSPCON2<0>）允许在接收过程中使能时钟延长。将 SEN 置 1 会导致在每个数据接收序列的末尾将 SCL 引脚保持在低电平。

19.4.4.1 7 位从动接收模式的时钟延长 （SEN = 1）

在 7 位从动接收模式下，如果在 $\overline{\text{ACK}}$ 序列末的第 9 个时钟的下降沿 BF 位被置 1，则 SSPCON1 寄存器中的 CKP 位就会自动清零，强制 SCL 输出保持在低电平。CKP 被清零会将 SCL 线拉为低电平。在允许继续接收之前，必须在用户中断服务程序中将 CKP 位置 1。保持 SCL线为低电平可以让用户在主器件发起另一个接收序列之前，有时间响应中断服务程序并读取SSPBUF的内容。这将防止发生缓冲器溢出 （见图 19-13）。

> 注 1：如果用户在第 9 个时钟的下降沿前读取了 SSPBUF 的内容，使得 BF 位被清零，那么 CKP 位就不会被清零，也不会发生时钟延长。
>
> 2：不管 BF 位的状态如何， CKP 位都可以用软件置 1。在下一个接收序列开始之前，用户需要在中断服务程序中清零 BF 位，以避免溢出。

19.4.4.2 10 位从动接收模式的时钟延长 （SEN = 1）

在 10 位从动接收模式下，在地址序列中会自动发生时钟延长，但是 CKP 位不会被清零。在这期间，如果 UA 位在第 9 个时钟之后置 1，就会启动时钟延长。UA 位在接收到 10 位地址的高字节和 10 位地址的第二个字节（R/$\overline{\text{W}}$ 位为零）之后，分别置 1。在更新 SSPADD 的过程中释放时钟线。如 7 位模式中描述的那样，在每个数据接收序列中都会发生时钟延长。

> 注： 如果用户在第9个时钟的下降沿出现之前查询 UA 位，并通过更新 SSPADD 寄存器清零 UA 位，而且用户没有提前读取 SSPBUF 寄存器使 BF 位清零，则 CKP 位的电平仍然不会被拉低。基于 BF 位状态的时钟延长仅在数据序列中出现，不会出现在地址序列中。

19.4.4.3 7 位从动发送模式的时钟延长

如果 BF 位被清零， 7 位从动发送模式将在第 9 个时钟的下降沿后清零 CKP 位，实现时钟延长。不管 SEN 位的状态如何，这种情况都会发生。

用户中断服务程序必须先将 CKP 位置 1 才可以继续发送。通过保持 SCL 线为低电平，用户在主器件发起另一个发送序列前，将有时间响应中断服务程序并装入 SSPBUF 的内容 （见图 19-9）。

> 注 1：如果用户在第 9 个时钟的下降沿到来之前就装入 SSPBUF 的内容，使 BF 位置 1。CKP 位就不会被清零，也不会发生时钟延长。
>
> 2：不管 BF 位的状态如何， CKP 位都可以用软件置 1。

19.4.4.4 10 位从动发送模式的时钟延长

在 10 位从动发送模式下，在前两个地址序列中由 UA 位的状态来控制时钟延长，正如在 10 位从动接收模式中一样。头两个地址后跟着第三个地址序列，该地址序列包含 10 位地址的高位和被置 1 的 R/$\overline{\text{W}}$ 位。在执行完第三个地址时序后，UA 位不置 1，此时模块配置为发送模式， BF 标志位控制时钟延长，就像在 7 位从动发送模式中一样 （见图 19-11）。

19.4.4.5 时钟同步和 CKP 位

当 CKP 位被清零时， SCL 输出被强制为 0。然而，将 CKP 位清零不会立即将 SCL 输出拉为低电平，除非已经采样到 SCL 输出为低电平，实际上需用外部 I^2C 主器件将 SCL 线拉低。SCL 输出将保持低电平，直到 CKP 位置 1 且 I^2C 总线上的其他器件将 SCL 电平拉高。这可以确保对 CKP 位的写操作不会违反 SCL 的最小高电平时间要求 （见图 19-12）。

图 19-12： 时钟同步时序

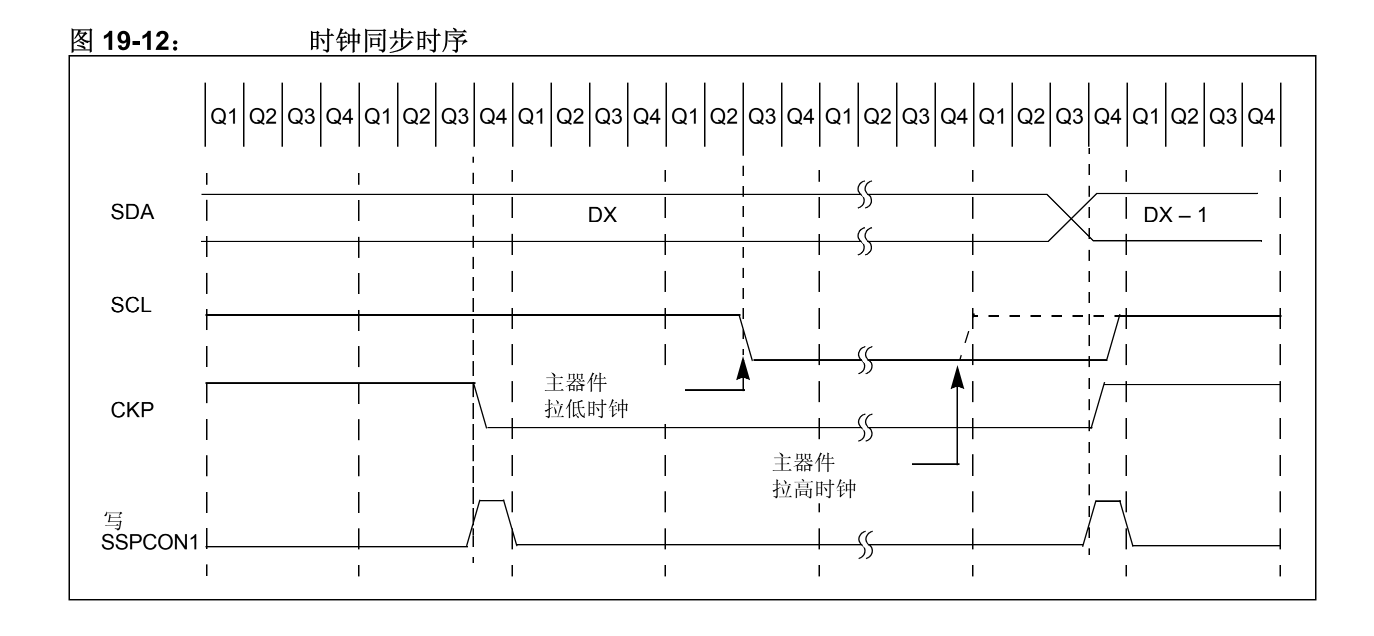
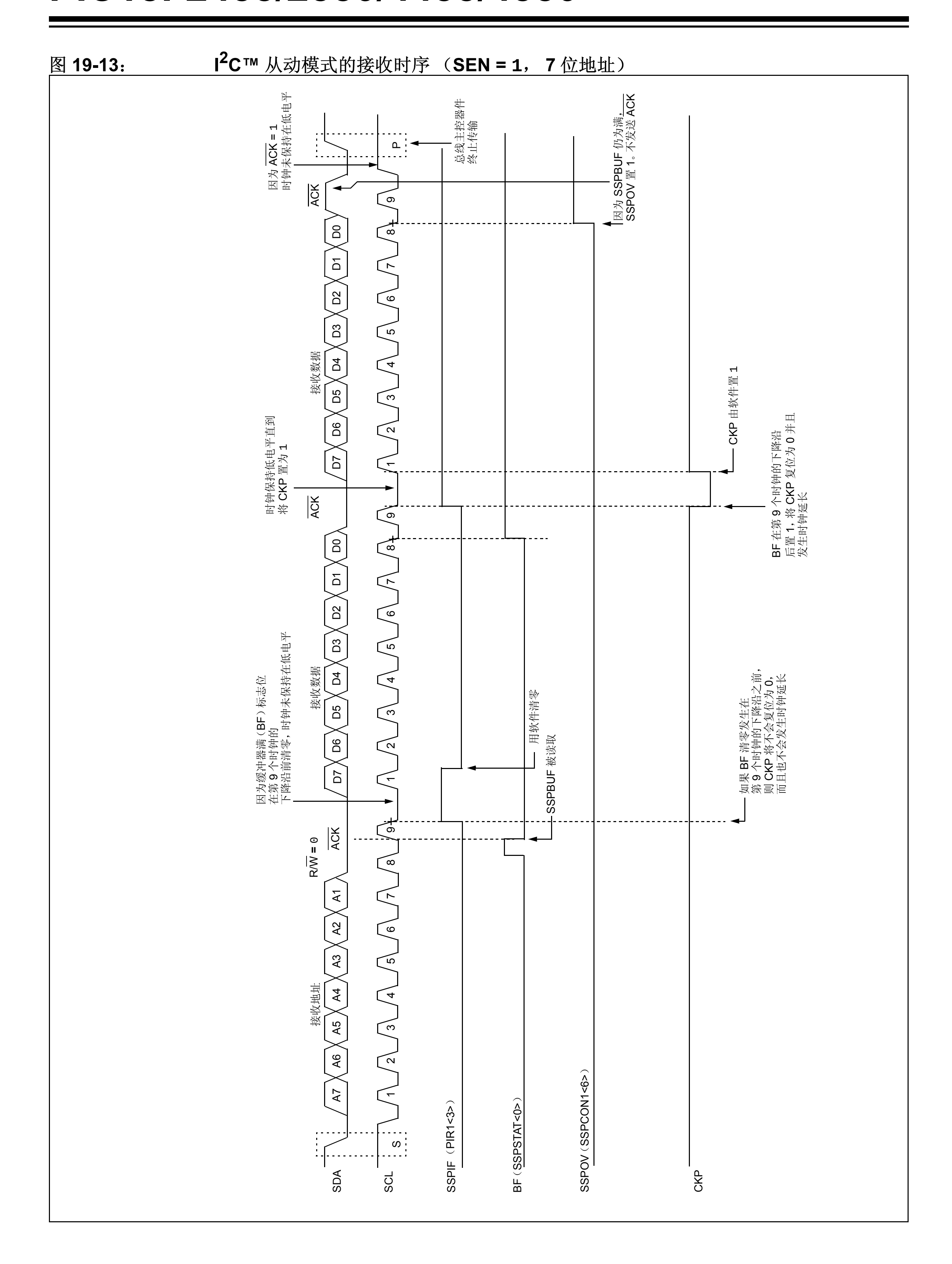
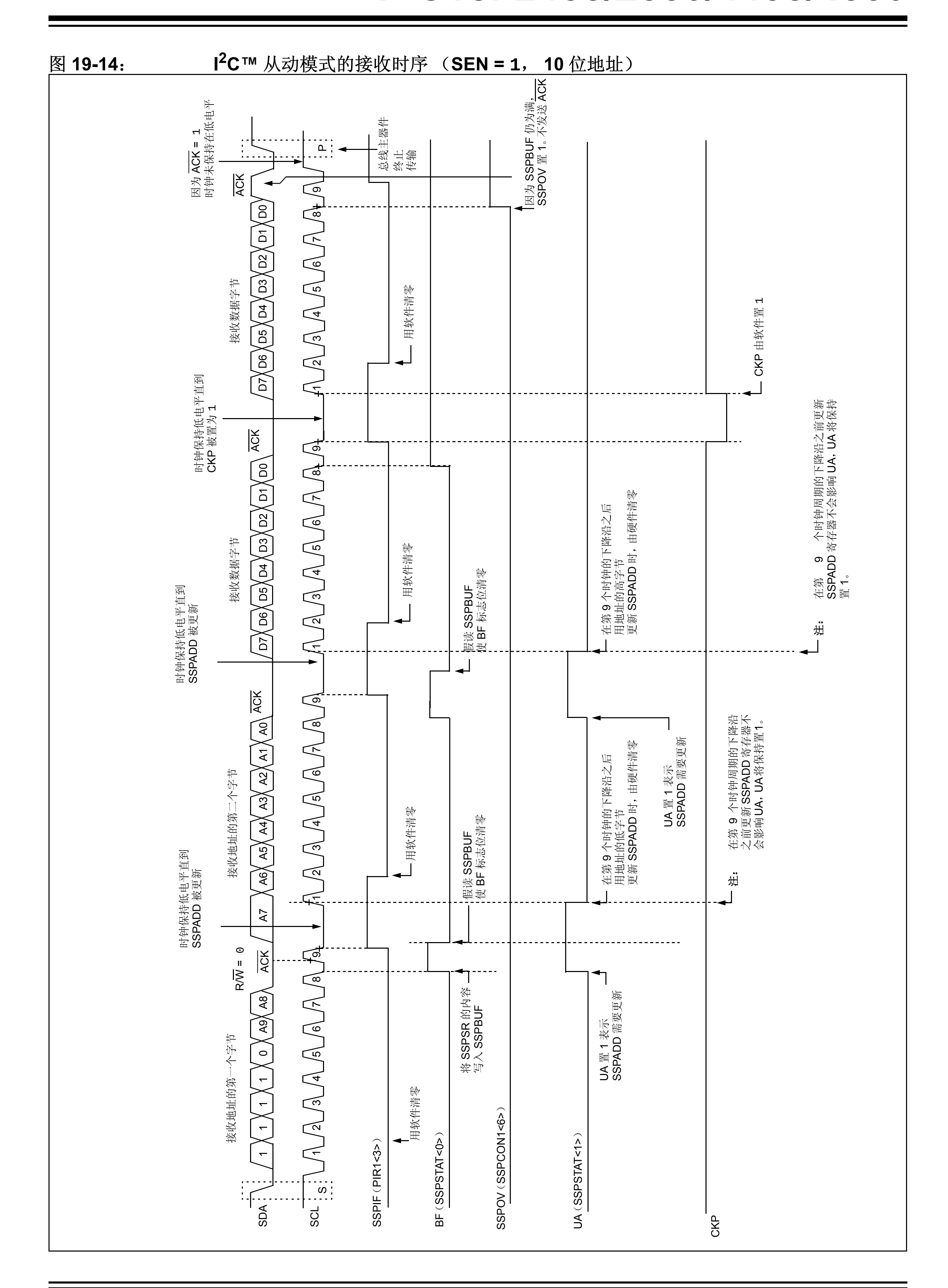


图 19-13： I²C™ 从动模式的接收时序 （SEN = 1， 7 位地址）



 初稿

图 19-14：　　I²C™ 从动模式的接收时序 （SEN = 1， 10 位地址）



19.4.5 广播呼叫地址支持

在 I²C 总线的寻址过程中，通常由启动条件后的第一个字节决定主器件将寻址哪个从器件。但广播呼叫寻址例外，它能寻址所有器件。当使用这种寻址方式时，理论上所有的器件都应该发送一个应答信号。

广播呼叫地址是由 I²C 协议为特定目的而保留的 8 个地址之一。它由全 0 组成，且 R/$\overline{W}$ = 0。

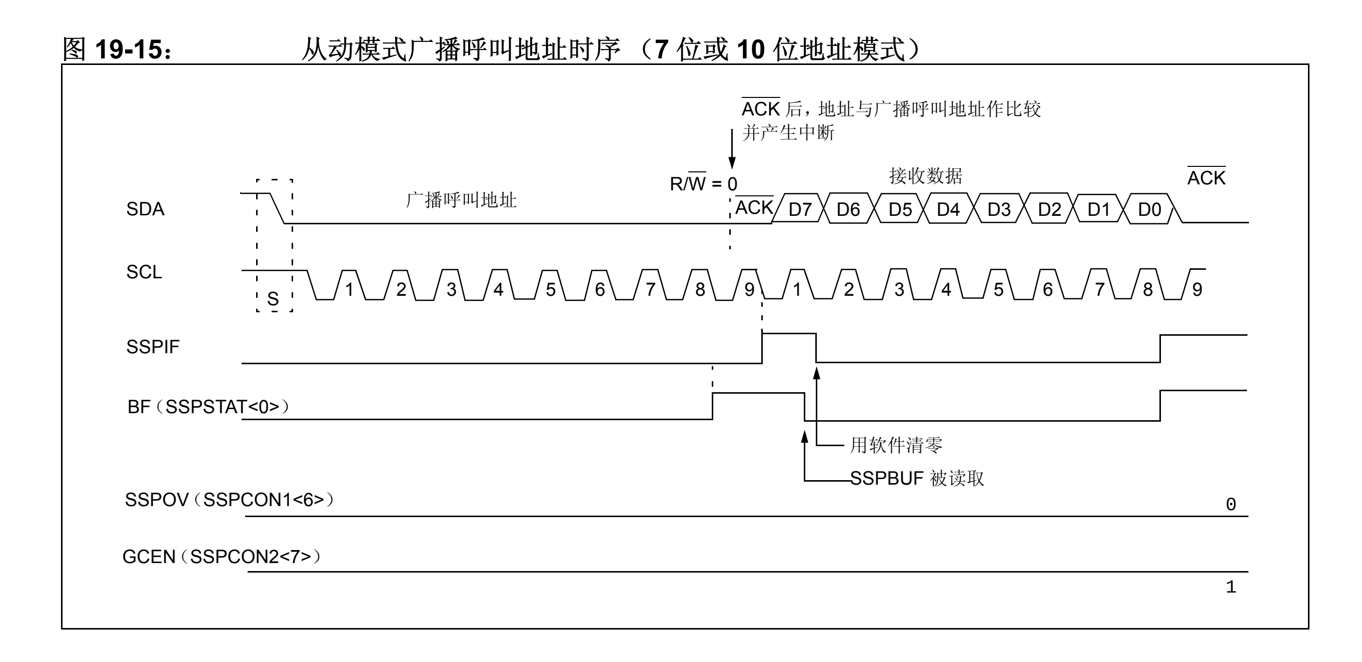
广播呼叫使能位（GCEN）（SSPCON2<7>）置 1 时，即可识别广播呼叫地址。检测到起始位后，8 位数据会移入 SSPSR，同时将该地址与 SSPADD 进行比较。它还会与广播呼叫地址进行比较并用硬件设定。

如果与广播呼叫地址匹配，SSPSR 的值将被传输到 SSPBUF，BF 标志位（第 8 位）置 1，并且在第 9 位（$\overline{ACK}$ 位）的下降沿将 SSPIF 中断标志位置 1。

当响应中断时，可以通过读取SSPBUF的内容来确定中断源。该值可以用于判断地址是特定器件的还是一个广播呼叫地址。

在 10 位模式下，需要更新 SSPADD 使之与地址的后半部分匹配，同时 UA 位（SSPSTAT<1>）置 1。如果 GCEN 位置 1 时采样到广播呼叫地址，同时从器件被配置为 10 位地址模式，则不再需要地址的后半部分，也不会将 UA 位置 1，从器件将在应答后开始接收数据（图 19-15）。

图 19-15：从动模式广播呼叫地址时序（7 位或 10 位地址模式）



 初稿

19.4.6 主控模式

通过将 SSPCON1 中相应的 SSPM 位置 1 和清零，同时将 SSPEN 位置 1，可以使能主控模式。在主控模式下， SCL 和 SDA 线由 MSSP 硬件控制。

主控模式的操作是通过在检测到启动和停止条件时产生中断来支持的。停止 （P）位和起始 （S）位在复位时或禁止 MSSP 模块时清零。当 P 位置 1 时，可以取得 I^2C 总线的控制权；否则总线处于空闲状态，且 P 位和 S 位都为零。

在固件控制的主控模式下，用户代码根据起始和停止位条件控制所有的 I^2C 总线操作。

一旦使能主控模式，用户即可选择以下 6 项操作：

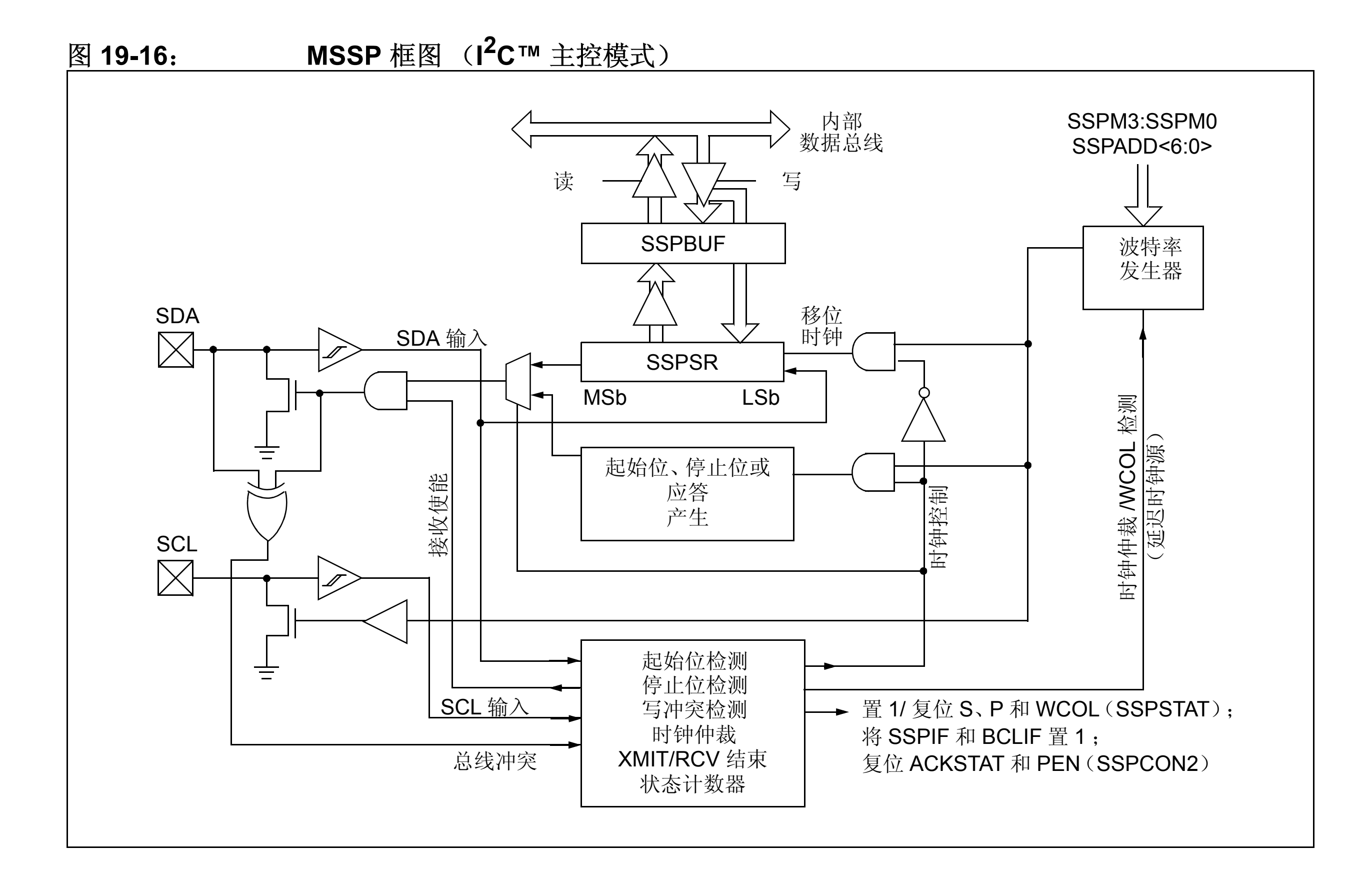
1. 在 SDA 和 SCL 上发出一个启动条件。
2. 在 SDA 和 SCL 上发出一个重复启动条件。
3. 写入 SSPBUF 寄存器，启动数据 / 地址的发送。
4. 配置 I^2C 端口用于接收数据。
5. 在接收到数据字节后产生应答条件。
6. 在 SDA 和 SCL 上产生停止条件。

注：	当被配置为 I^2C 主控模式时，MSSP 模块不允许事件排队。例如，不允许用户在启动条件尚未完成前立即写 SSPBUF 寄存器以启动传输。这种情况下，将不会写入 SSPBUF，WCOL 位将被置 1，这表明没有发生对 SSBUF 的写操作。

下列事件会将 SSP 中断标志位 SSPIF 置 1（如果允许，将产生 SSP 中断）：

- 启动条件
- 停止条件
- 数据传输字节已发送 / 已接收
- 应答发送
- 重复启动

图 19-16： MSSP 框图 （I^2C™ 主控模式）



19.4.6.1 I²C 主控模式工作原理

主器件产生所有串行时钟脉冲和启动 / 停止条件，以停止或重复启动条件结束传输。因为重复启动条件也是下一次串行传输的开始，因此不会释放 I²C 总线。

在主控发送器模式下，串行数据通过 SDA 输出，而串行时钟由 SCL 输出。发送的第一个字节包括接收器件的地址（7 位）和读 / 写（R/W）位。在这种情况下， R/W 位将是逻辑 0。一次发送 8 位串行数据。每发送一个字节，会收到一个应答位。启动和停止条件表明串行传输的开始和结束。

在主控接收模式下，发送的第一个字节包括发送器件的地址 （7 位）和 R/W 位。在这种情况下， R/W 位将是逻辑 1。因此，发送的第一个字节是一个 7 位从器件地址，后面跟 1 表示接收。串行数据通过 SDA 接收，而串行时钟由 SCL 输出。每次接收 8 位串行数据。每接收到一个字节，都会发送一个应答位。启动和停止条件分别表明发送的开始和结束。

在 I²C 模式下，在 SPI 模式中使用的波特率发生器被用于将 SCLx 时钟频率设置为 100 kHz、 400 kHz 或 1 MHz。详情请参见第 **19.4.7 节 “波特率”**。

下面是一个典型的发送序列：

1. 用户通过将启动使能位 SEN （SSPCON2<0>）置 1 产生启动条件。
2. SSPIF 已置 1。在进行下一步操作前，MSSP 模块将等待所需的启动时间。
3. 用户将从器件地址装入 SSPBUF 进行发送。
4. 地址从 SDA 引脚移出，直到发送完所有 8 位为止。
5. MSSP 模块移入来自从器件的 ACK 位，并将它的值写入 SSPCON2 寄存器（SSPCON2<6>）。
6. MSSP 模块在第 9 个时钟周期的末尾将 SSPIF 置 1，产生一个中断。
7. 用户将 8 位数据装入 SSPBUF。
8. 数据从 SDA 引脚移出，直到发送完所有 8 位为止。
9. MSSP 模块移入来自从器件的 ACK 位，并将它的值写入 SSPCON2 寄存器（SSPCON2<6>）。
10. MSSP 模块在第 9 个时钟周期的末尾将 SSPIF 置 1，产生一个中断。
11. 用户通过将停止使能位 PEN （SSPCON2<2>）置 1 产生停止条件。
12. 一旦停止条件完成，将产生一个中断。

19.4.7 波特率

在 I²C 主控模式下，波特率发生器（Baud Rate Generator，BRG）的重载值位于 SSPADD 寄存器（图 19-17）的低 7 位。当写入 SSPBUF 时，波特率发生器将自动开始计数。BRG 会递减计数至 0，然后停止直到再次装入重载值。BRG 计数器会在每个指令周期（TCY）中的 Q2 和 Q4 时钟周期上进行两次减计数。在 I²C 主控模式下，会自动重载 BRG。

如果指定操作完成（即在传输的最后一个数据位后面跟着 ACK），内部时钟将自动停止计数，SCL 引脚将保持在其最后的状态。

表 19-3 给出了不同的指令周期下的时钟频率以及装入 SSPADD 的 BRG 值。

图 19-17： 波特率发生器框图

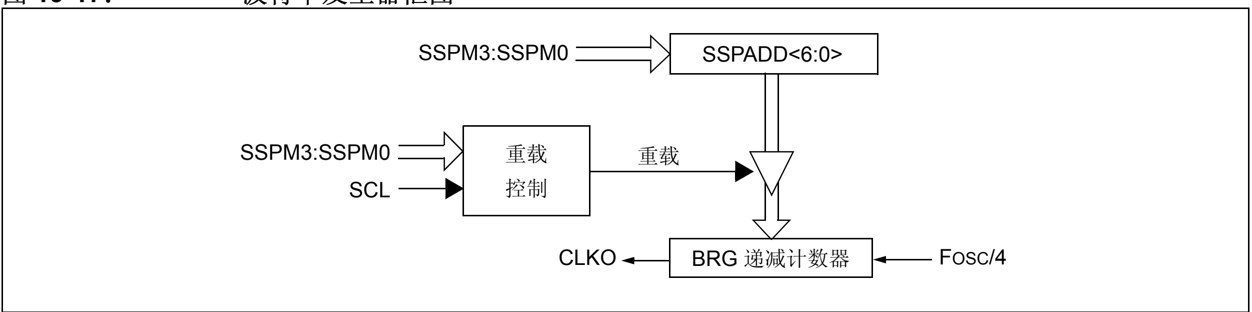


表 19-3： 带有 BRG 的 I²C™ 时钟速率

FCY	FCY * 2	BRG 值	FSCL（两次 BRG 计满返回）
10 MHz	20 MHz	19h	400 kHz(1)
10 MHz	20 MHz	20h	312.5 kHz
10 MHz	20 MHz	3Fh	100 kHz
4 MHz	8 MHz	0Ah	400 kHz(1)
4 MHz	8 MHz	0Dh	308 kHz
4 MHz	8 MHz	28h	100 kHz
1 MHz	2 MHz	03h	333 kHz(1)
1 MHz	2 MHz	0Ah	100 kHz
1 MHz	2 MHz	00h	1 MHz(1)

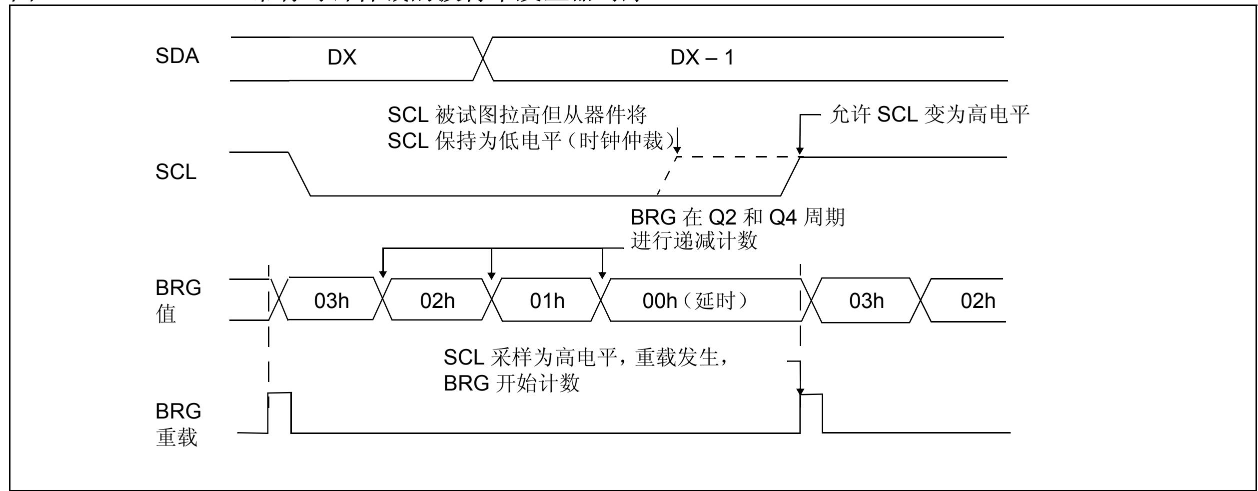
注 1：虽然 I²C™ 接口各方面都不符合 400 kHz I²C 规范（该规范适用于大于 100 kHz 的频率），但在需要较高频率的应用场合可以慎重使用。

 初稿

19.4.7.1 时钟仲裁

如果在任何接收、发送或重复启动 / 停止条件过程中，主器件拉高了 SCL 引脚（允许 SCL 引脚悬空为高电平），就会发生时钟仲裁。如果允许 SCL 引脚悬空为高电平，波特率发生器（BRG）将暂停计数直到实际采样到 SCL 引脚为高电平为止。当 SCL 引脚被采样为高电平时，会将 SSPADD<6:0> 的内容重新装入波特率发生器并使之开始计数。这可以保证当外部器件将时钟拉低时，SCL 始终至少保持一个 BRG 计满返回周期的高电平（图 19-18）。

图 19-18： 带有时钟仲裁的波特率发生器时序



19.4.8 I²C 主控模式启动条件时序

要产生启动条件，用户应将启动使能位 SEN（SSPCON2<0>）置 1。当 SDA 和 SCL 引脚采样为高电平时，波特率发生器重新装入 SSPADD<6:0> 的值并开始计数。当波特率发生器发生超时 （TBRG）时，如果 SCL 和 SDA 都被采样为高电平，则 SDA 引脚被驱动为低电平。当 SCL 为高电平时，将 SDA 驱动为低电平会产生启动条件，并使 S 位（SSPSTAT<3>）置 1。随后波特率发生器再次装入 SSPADD<6:0> 的值并恢复计数。当波特率发生器再次发生超时（TBRG）时，SEN 位（SSPCON2<0>）将被硬件自动清零，波特率发生器暂停工作，SDA 信号保持为低电平，启动条件完成。

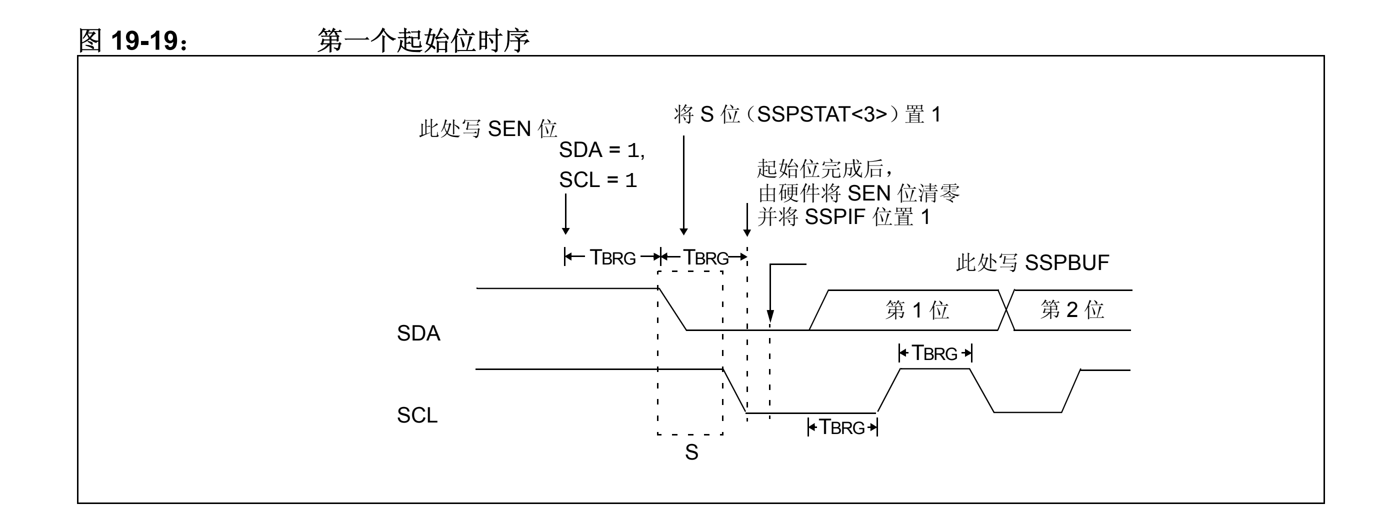
注：	如果在启动条件开始时，SDA 和 SCL 引脚已经被采样为低电平，或者在启动条件期间，SCL 在 SDA 线被驱动为低电平之前已经采样为低电平，则会产生总线冲突。总线冲突中断标志位 BCLIF 置 1，启动条件中止，I²C 模块复位到空闲状态。

19.4.8.1 WCOL 状态标志

当启动时序进行时，如果用户写 SSPBUF，则 WCOL 被置 1，同时缓冲器内容不变 （写操作无效）。

注：	由于不允许事件排队，在启动条件完成之前，不能对 SSPCON2 的低 5 位进行写操作。

图 19-19： 第一个起始位时序



19.4.9 I2C 主控模式重复启动条件时序

将 RSEN 位（SSPCON2<1>）编程为高电平，并且 I2C 逻辑模块处于空闲状态时，就会产生重复启动条件。当 RSEN 位置 1 时， SCL 引脚被拉为低电平。当 SCL 引脚被采样为低电平时，波特率发生器装入 SSPADD<6:0> 的值，并开始计数。在一个波特率发生器计数周期（TBRG）内，SDA 引脚被释放（其引脚电平被拉高）。当波特率发生器超时时，如果 SDA 被采样为高电平， SCL 引脚将被拉高。当 SCL 被采样为高电平时，波特率发生器重新装入 SSPADD<6:0> 的值并开始计数。SDA 和 SCL 必须在该 TBRG 内采样为高电平。接下来，在下一个 TBRG 中，SCL 保持高电平，SDA 引脚被驱动为低电平（SDA = 0）。然后 RSEN 位（SSPCON2<1>）将自动清零，波特率发生器不会被重载并且 SDA 引脚保持低电平。一旦在 SDA 和 SCL 引脚上检测到启动条件，S 位（SSPSTAT<3>）就将被置 1。直到波特率发生器发生超时后，SSPIF 位才会置 1。

注	
	1： 有其他事件在进行时，编程设置对 RSEN 无效。 **2：** 在重复启动条件发生期间，下列事件将会导致总线冲突： • 当 SCL 由低电平变为高电平时，采样到 SDA 为低电平。 • 在 SDA 被拉低之前， SCL 变为低电平。这表示另一个主器件正尝试发送数据 1。

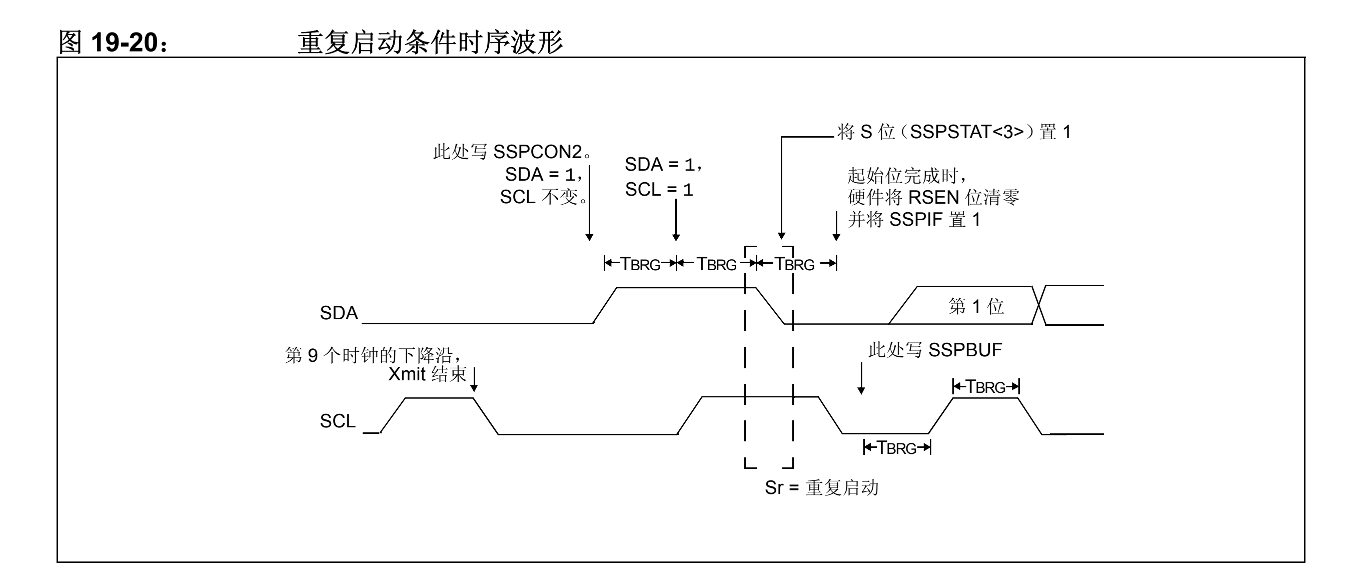
一旦 SSPIF 位被置 1，用户便可以在 7 位地址模式下将 7 位地址，或者在 10 位地址模式下将默认的第一个地址字节写入 SSPBUF。当发送完第一个 8 位字节并接收到一个 ACK 后，用户可以发送另外 8 位地址（10 位地址模式）或 8 位数据（7 位地址模式）。

19.4.9.1 WCOL 状态标志

在重复启动时序进行当中，如果用户写 SSPBUF，则 WCOL 被置 1，同时缓冲器内容不变（写操作无效）。

注：	由于不允许事件排队，在重复启动条件完成之前，不能对 SSPCON2 的低 5 位进行写操作。

图 19-20： 重复启动条件时序波形



 初稿

19.4.10 I²C 主控模式发送

发送一个数据字节、一个 7 位地址或一个 10 位地址的一半，都是通过写一个值到 SSPBUF 寄存器来实现的。该操作将使缓冲器满标志位 BF 置 1，波特率发生器开始计数，同时启动下一次发送。在出现 SCL 的下降沿后（见表 28-20 中，数据保持时间规范参数 106），地址 / 数据的每一位被移出至 SDA 引脚。在一个波特率发生器计满返回周期（TBRG）内，SCL 保持低电平。数据应该在 SCL 释放为高电平前保持有效（见表 28-20 中，数据建立时间规范参数 107）。当 SCL 引脚被释放为高电平时，它将在一个 TBRG 内保持为高电平。在此期间以及 SCL 的下一个下降沿之后的一段时间内，SDA 引脚上的数据必须保持稳定。在第 8 位数据被移出（第 8 个时钟周期的下降沿）之后，BF 标志位被清零，同时主器件释放 SDA。此时如果发生地址匹配或是数据被正确接收，被寻址的从器件将在第 9 位时以一个 $\overline{\text{ACK}}$ 位做出响应。$\overline{\text{ACK}}$ 的状态在第 9 个时钟周期的下降沿写入 ACKDT 位。主器件接收到应答之后，应答状态位 ACKSTAT 会被清零。如果未收到应答，则该位保持置 1。第 9 个时钟周期之后，SSPIF 位会置 1，主控时钟（波特率发生器）暂停，直到下一个数据字节装入 SSPBUF。SCL 引脚保持低电平并且 SDA 保持不变（图 19-21）。

在写 SSPBUF 之后，地址的每一位在 SCL 的下降沿被移出，直至所有 7 位地址和 R/$\overline{\text{W}}$ 位都被移出。在第 8 个时钟的下降沿，主器件将 SDA 引脚拉为高电平，以允许从器件发出一个应答响应。在第 9 个时钟的下降沿，主器件通过采样 SDA 引脚来判断地址是否被从器件识别。$\overline{\text{ACK}}$ 位的状态被装入 ACKSTAT 状态位（SSPCON2<6>）。在发送地址的第 9 个时钟下降沿之后，SSPIF 置 1，BF 标志位清零，波特率发生器关闭直到下一次写 SSPBUF，同时 SCL 引脚保持低电平并且允许 SDA 引脚悬空。

19.4.10.1 BF 状态标志

在发送模式下，BF 位（SSPSTAT<0>）在 CPU 写 SSPBUF 时置 1，在所有 8 位数据移出后清零。

19.4.10.2 WCOL 状态标志

如果用户在发送过程中（即 SSPSR 仍在移出数据字节时）写 SSPBUF，则 WCOL 置 1 并且缓冲器内容不变（写操作无效）。

WCOL 必须用软件清零。

19.4.10.3 ACKSTAT 状态标志

在发送模式下，当从器件发送应答（$\overline{\text{ACK}}$ = 0）时，ACKSTAT 位（SSPCON2<6>）清零；当从器件没有应答（$\overline{\text{ACK}}$ = 1）时，该位置 1。从器件在识别出其地址（包括广播呼叫地址）或正确接收数据后，会发出一个应答。

19.4.11 I²C 主控模式接收

通过编程接收使能位 RCEN（SSPCON2<3>）使能主控模式接收。

注：	RCEN 位置 1 前，MSSP 模块必须处于空闲状态，否则上述操作无效。

波特率发生器开始计数，每次计满返回时，SCL 引脚的状态就会发生改变（由高变低或由低变高），同时数据被移入 SSPSR。第 8 个时钟的下降沿之后，接收使能标志位自动清零，SSPSR 的内容被装入 SSPBUF，BF 标志位置 1，SSPIF 标志位置 1 并且波特率发生器暂停计数，且 SCL 保持为低电平。此时 MSSP 处于空闲状态，等待下一条命令。当 CPU 读缓冲器时，BF 标志位将自动清零。通过将应答序列使能位 ACKEN（SSPCON2<4>）置 1，用户可以在接收结束后发送应答位。

19.4.11.1 BF 状态标志

接收数据过程中，把地址或数据字节从 SSPSR 装入 SSPBUF 时，BF 位置 1；在读 SSPBUF 寄存器时将其清零。

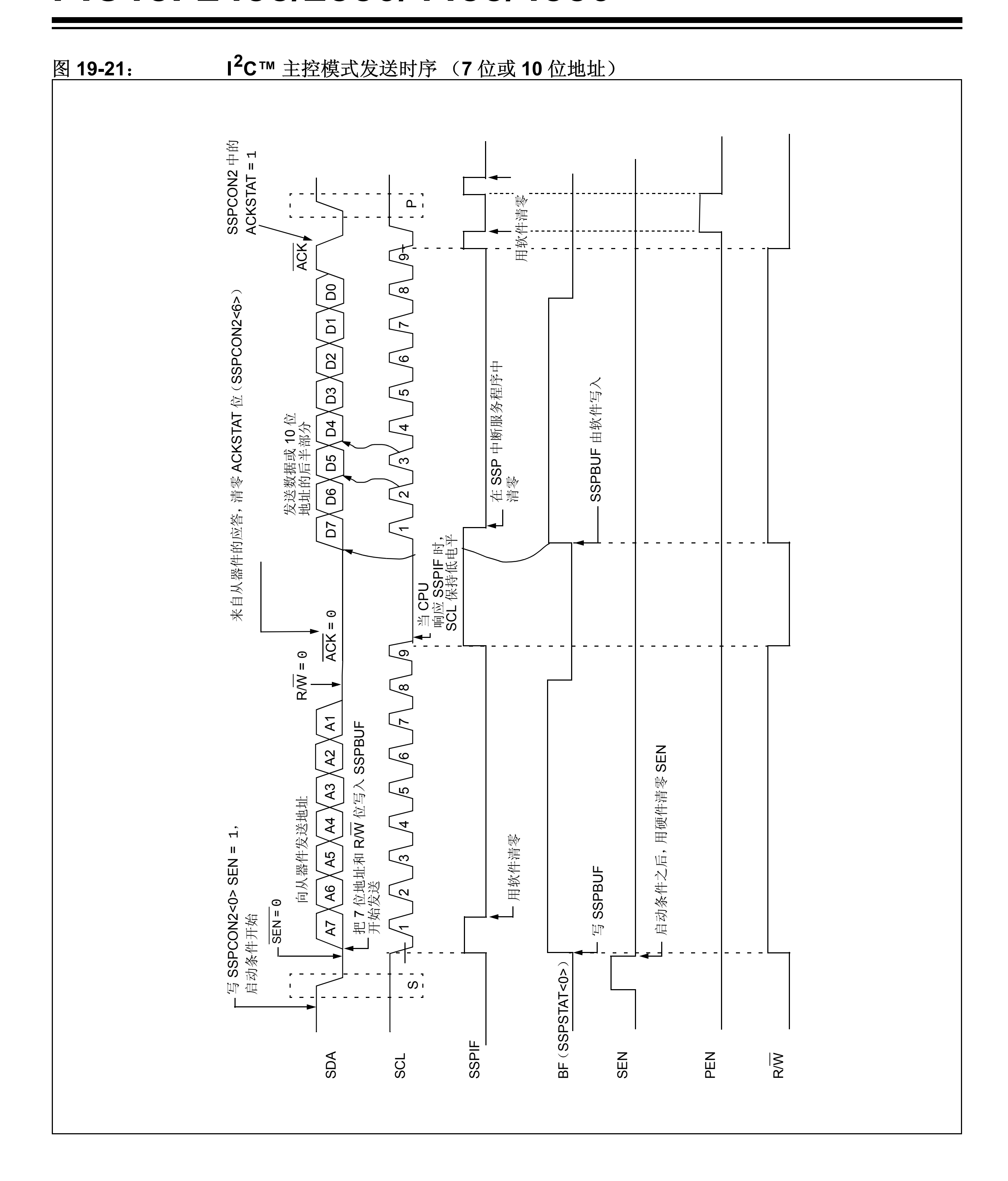
19.4.11.2 SSPOV 状态标志

接收数据过程中，当SSPSR接收到8位数据时，SSPOV 位置 1，而 BF 标志位已经在上一次接收时就置 1 了。

19.4.11.3 WCOL 状态标志

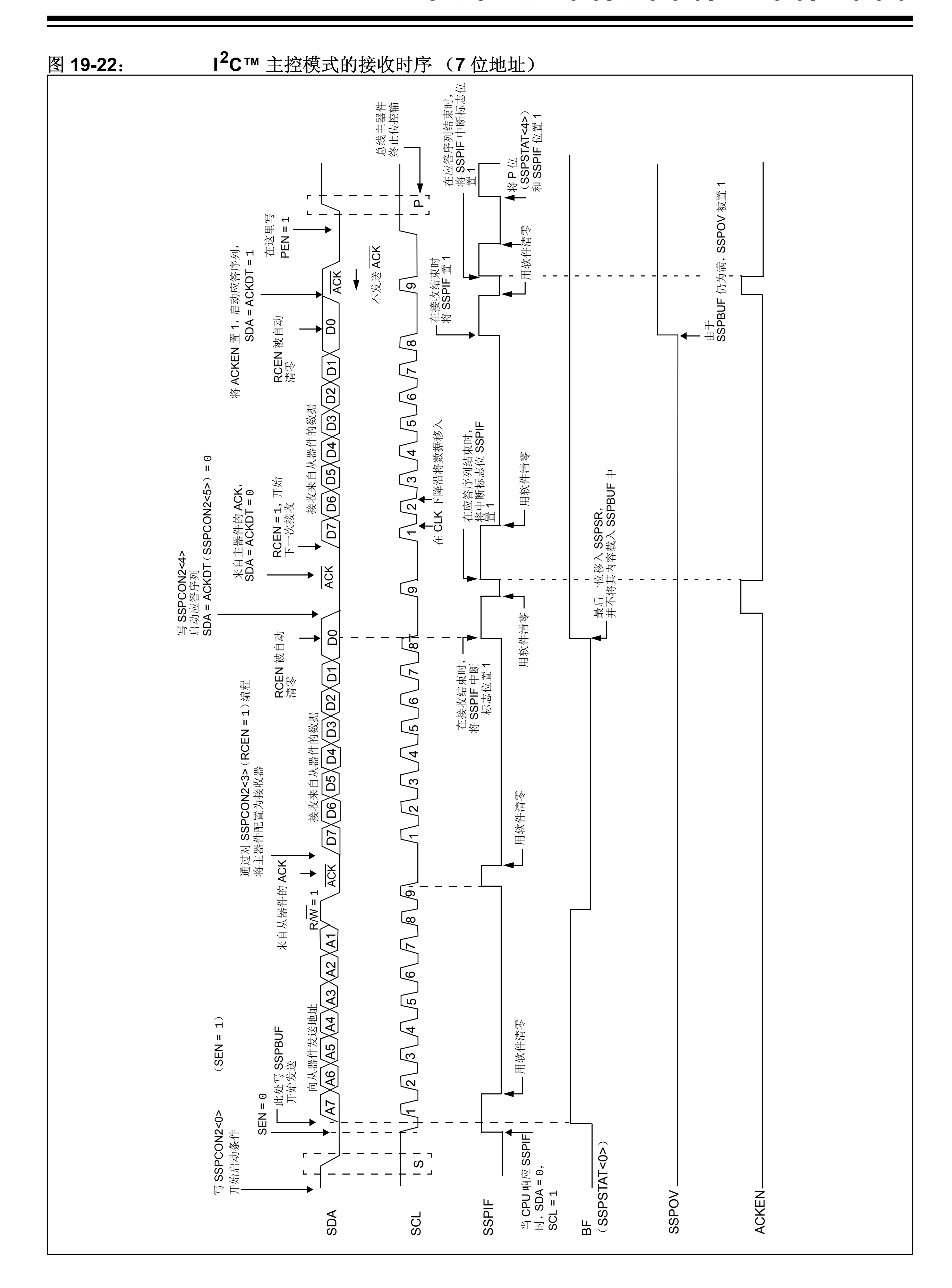
如果用户在接收过程中（即 SSPSR 仍在移入数据字节时）写 SSPBUF，则 WCOL 置 1 并且缓冲器内容不变（写操作无效）。

图 19-21： **I²C™ 主控模式发送时序 （7 位或 10 位地址）**



 初稿

图 19-22：　　I^2C™ 主控模式的接收时序 （7 位地址）



19.4.12 应答序列时序

将应答序列使能位 ACKEN（SSPCON2<4>）置 1 即可使能应答序列。当该位被置 1 时， SCL 引脚被拉低，应答数据位的内容出现在 SDA 引脚上。如果用户希望产生一个应答，则应该将 ACKDT 位清零。如果不希望产生应答，用户应当在应答序列开始前将 ACKDT 位置 1。然后波特率发生器进行一个计满返回周期 （TBRG）的计数，SCL 引脚电平被拉高。当 SCL 引脚被采样为高电平时 （时钟仲裁），波特率发生器再进行一个 TBRG 周期的计数。然后 SCL 引脚被拉低。在这之后， ACKEN 位自动清零，波特率发生器关闭并且 MSSP 模块进入空闲模式 （图 19-23）。

19.4.12.1 WCOL 状态标志

如果用户在应答序列进行过程中试图写 SSPBUF，则 WCOL 将置 1 并且缓冲器的内容不会改变 （写操作无效）。

19.4.13 停止条件时序

将停止序列使能位 PEN （SSPCON2<2>）置 1，在接收 / 发送结束后， SDA 引脚上产生停止位。在接收 / 发送结束时，SCL 引脚在第 9 个时钟的下降沿后保持低电平。当 PEN 位置 1 时，主器件将 SDA 线置为低电平。当 SDA 线被采样为低电平时，波特率发生器被重载并递减计数至 0。当波特率发生器超时时，SCL 引脚被拉为高电平，在一个 TBRG（波特率发生器计满返回周期）之后， SDA 引脚将被拉高。当 SDA 引脚被采样为高电平且 SCL 也是高电平时，P 位（SSPSTAT<4>）置 1。在一个 TBRG 之后， PEN 位清零，同时 SSPIF 位置 1 （图 19-24）。

19.4.13.1 WCOL 状态标志

如果用户在停止序列进行过程中试图写 SSPBUF，则 WCOL位将置1并且缓冲器的内容不变（写操作无效）。

图 19-23： 应答序列波形

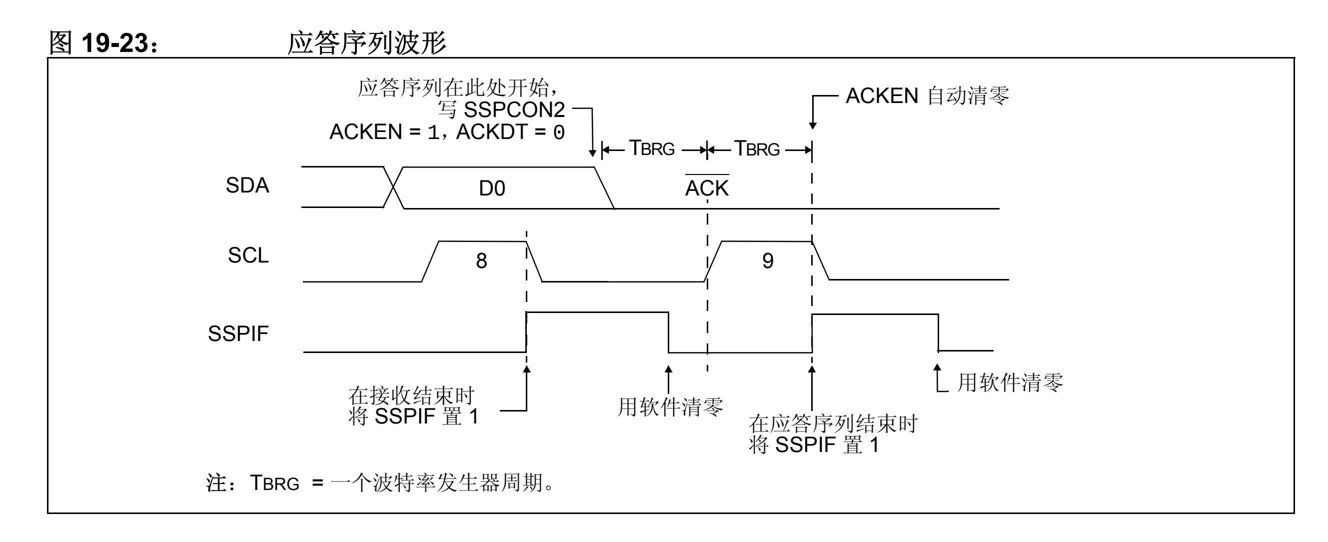
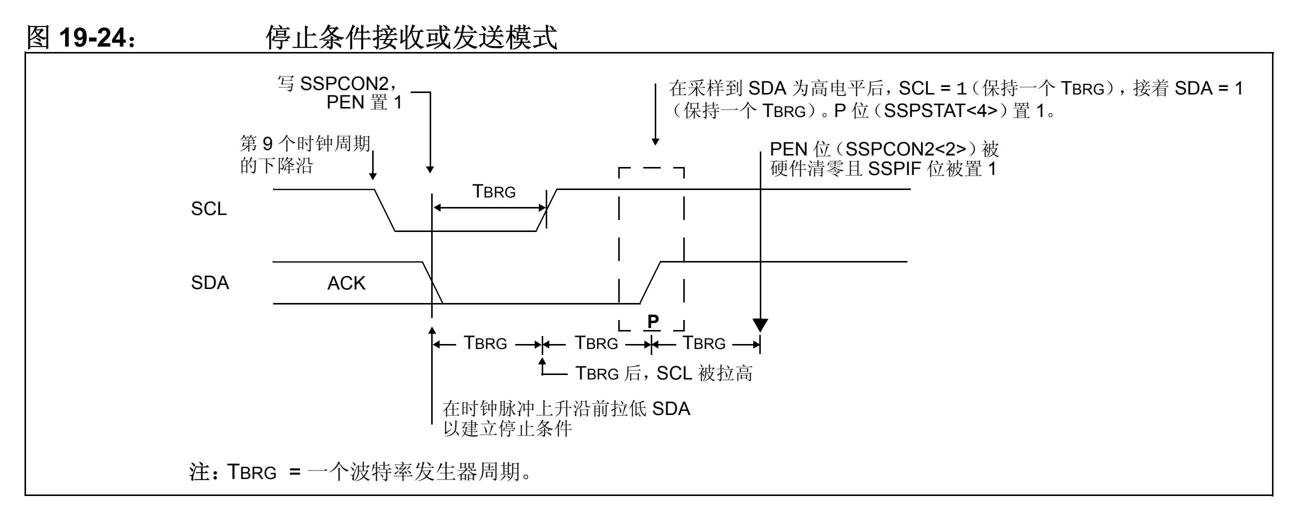


图 19-24： 停止条件接收或发送模式



 初稿

19.4.14 休眠模式下的工作方式

在休眠模式下， I²C 模块能够接收地址或数据。并且在地址匹配或字节传输完成后，如果允许 MSSP 中断，将唤醒处理器。

19.4.15 复位的影响

复位会禁止 MSSP 模块并终止当前的传输。

19.4.16 多主机模式

在多主机模式下，在检测到启动和停止条件时将产生中断，这可以用于判断总线何时空闲。停止 （P）位和起始（S）位在复位时或禁止 MSSP 模块时清零。当 P 位（SSPSTAT<4>）置 1 时，可以取得 I²C 总线的控制权；否则总线处于空闲状态，且 P 位和 S 位都为零。当总线忙时，如果出现停止条件，允许 SSP 中断的情况下将产生中断。

在多主机工作模式下，必须监视 SDA 线来进行仲裁，查看信号电平是否为期望的输出电平。此操作由硬件实现，其结果保存在 BCLIF 位中。

可能导致仲裁失败的情况是：

- 地址传输
- 数据传输
- 启动条件
- 重复启动条件
- 应答条件

19.4.17 多主机通信、总线冲突与总线仲裁

多主机模式是通过总线仲裁来支持的。当主器件将地址 / 数据位输出到 SDA 引脚时，如果一个主器件在 SDA 上输出 1，而另一个主器件输出 0，就会发生总线仲裁。如果 SDA 引脚上期望的数据是 1，而实际采样到的数据是 0，则发生了总线冲突。主器件将把总线冲突中断标志位 BCLIF 置 1，并将 I²C 端口复位到空闲状态（图 19-25）。

如果在发送过程中发生总线冲突，则会暂停发送，并且 BF 标志位被清零， SDA 和 SCL 线被拉高同时将 SSPBUF 置于可写入状态。当执行完总线冲突中断服务程序后，如果 I²C 总线空闲，用户可通过发出启动条件恢复通信。

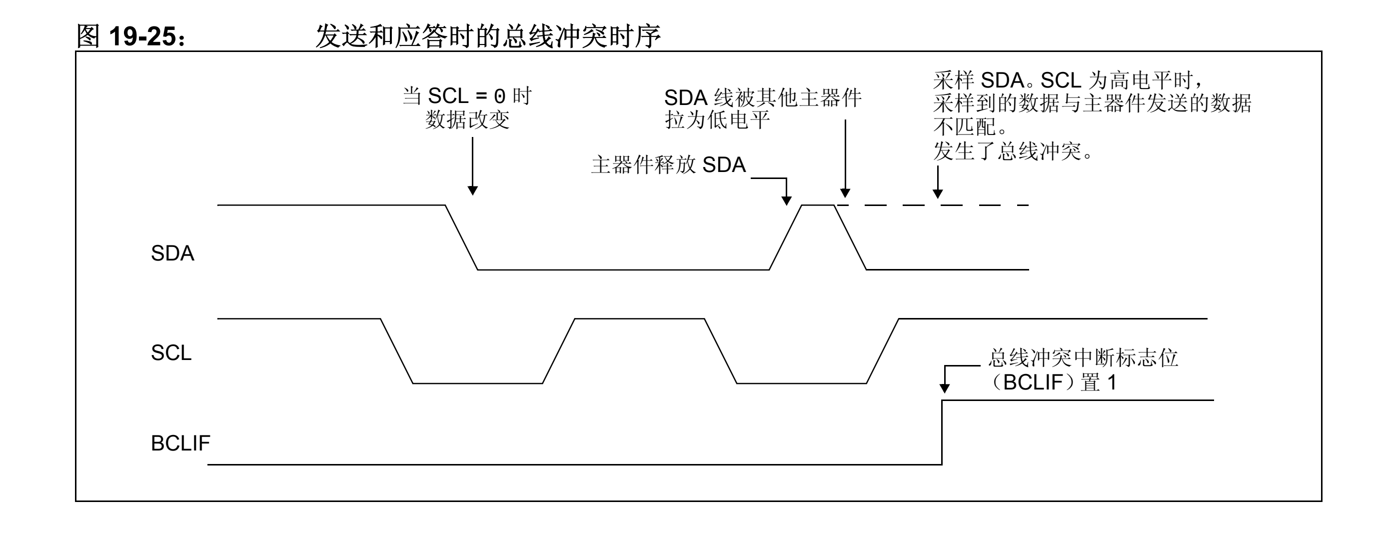
如果在启动、重复启动、停止或应答条件进行的过程中发生总线冲突，那么上述条件将被中止，SDA 和 SCL 线被拉高并且 SSPCON2 寄存器中的对应控制位清零。当执行完总线冲突中断服务程序后，如果 I²C 总线空闲，用户可通过发出启动条件恢复通信。

主器件将继续监视 SDA 和 SCL 引脚。一旦产生停止条件， SSPIF 位将被置 1。

发生总线冲突时，无论发送的进度如何，写入 SSPBUF 都会重新从第一个数据位开始发送数据。

在多主机模式下，通过在检测到启动和停止条件时产生中断可以确定总线何时空闲。 SSPSTAT 寄存器中的 P 位置 1 时，可以取得 I²C 总线的控制权；否则总线处于空闲状态，且 P 位和 S 位均为零。

图 19-25： 发送和应答时的总线冲突时序



19.4.17.1 启动条件期间的总线冲突

启动条件期间，以下事件将导致总线冲突：

a) 在启动条件开始时， SDA 或 SCL 被采样为低电平 （图 19-26）。
b) SDA被拉低之前，SCL被采样为低电平（图 19-27）。

在启动条件期间， SDA 和 SCL 引脚都会被监视。

如果 SDA 引脚已经是低电平，或 SCL 引脚已经是低电平，则：

- 中止启动条件，
- BCLIF 标志位置 1，
- MSSP 模块复位为空闲状态 （图 19-26）。

启动条件从 SDA 和 SCL 引脚被拉高开始。当 SDA 引脚被采样为高电平时，波特率发生器装入 SSPADD<6:0> 的内容并递减至 0。如果在 SDA 为高电平时，采样到 SCL 引脚为低电平，则发生了总线冲突，因为这表示另一台主器件在启动条件期间正试图发送一个数据 1。

如果 SDA 引脚在该计数周期内采样为低电平，则 BRG 复位，同时 SDA 线保持原值（图 19-28）。但是，如果 SDA 引脚被采样为 1，SDA 引脚将在 BRG 计数结束时被置为低电平。接着，重载波特率发生器并使之递减至 0，在此期间，如果采样到 SCL 引脚为 0，则不会发生总线冲突。在BRG计数结束时，SCL引脚被拉为低电平。

> **注：** 在启动条件期间不太可能发生总线冲突，因为两个总线主控器件不可能精确地在同一时刻发出启动条件。因此一个主器件将总是先于另一个主器件将 SDA 拉低。由于必须允许两个主器件对启动条件后的第一个地址进行仲裁，因此上述情况不会引起总线冲突。如果地址是相同的，必须继续对数据部分、重复启动或停止条件进行仲裁。

图 19-26： 启动条件期间的总线冲突 （仅 SDA）

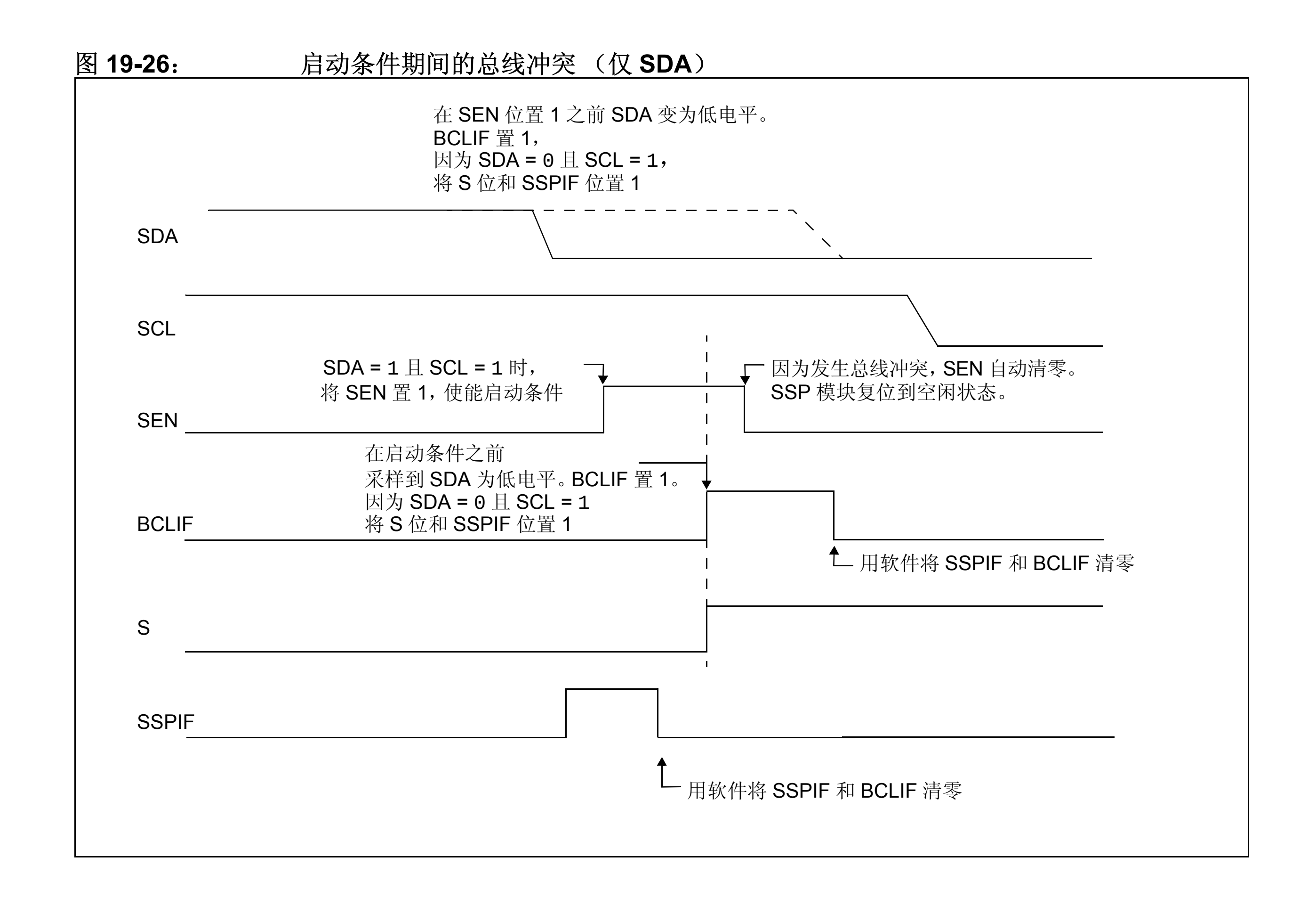


图 19-27： 启动条件期间的总线冲突 （SCL = 0）

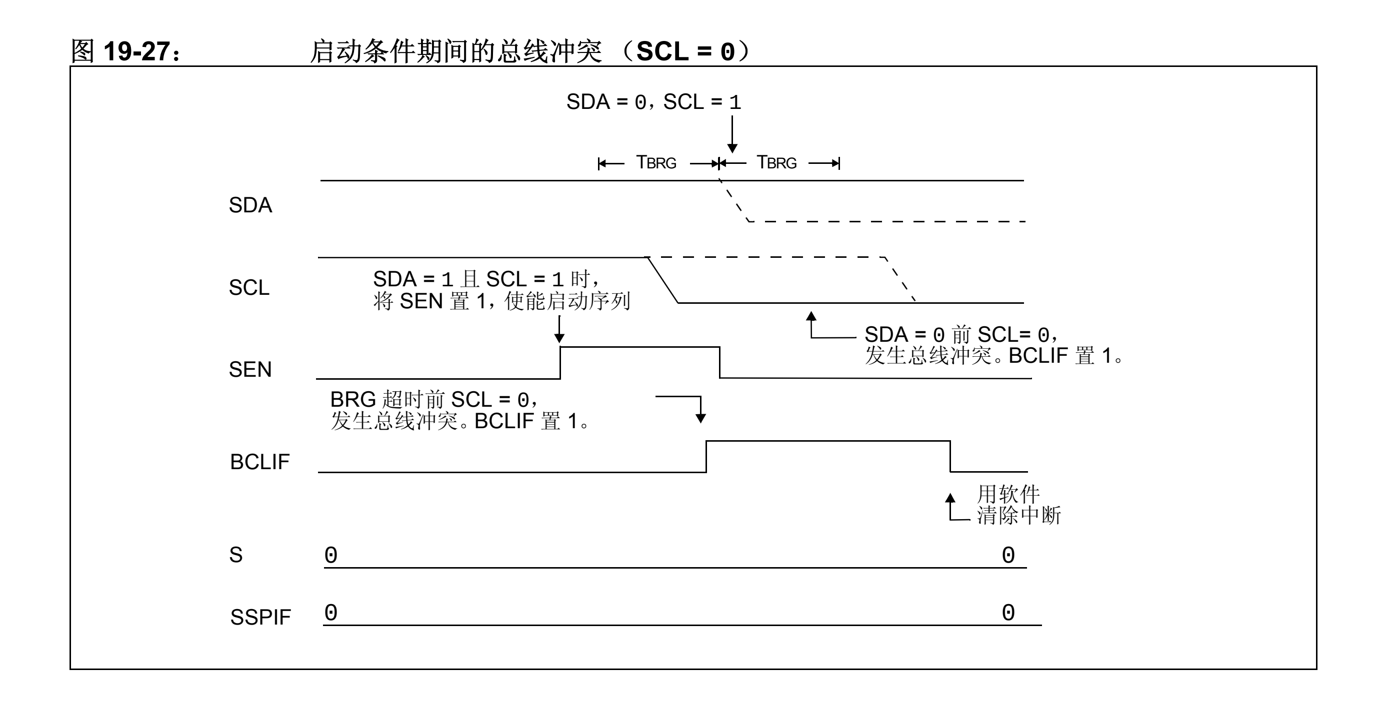
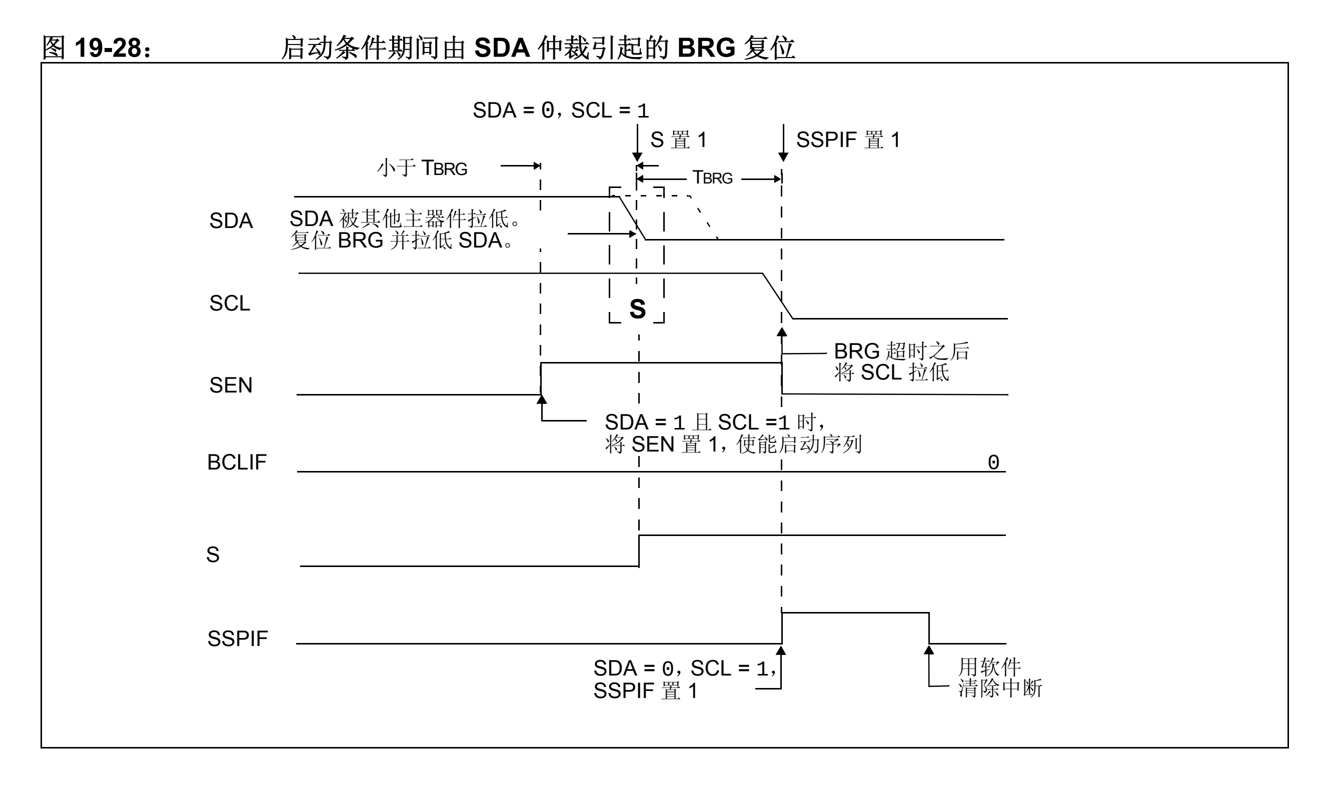


图 19-28： 启动条件期间由 SDA 仲裁引起的 BRG 复位



19.4.17.2 重复启动条件期间的总线冲突

在重复启动条件期间，以下事件会导致发生总线冲突：

a) 在 SCL 由低电平变为高电平的过程中，在 SDA 上采样到低电平。
b) 在 SDA 被拉为低电平之前， SCL 变为低电平，表示另一个主器件正试图发送数据 1。

当用户拉高 SDA 并允许该引脚悬空时， BRG 装入 SSPADD<6:0> 中的值并递减计数至 0，接着 SCL 引脚被置为高电平，当 SCL 采样到高电平时，对 SDA 引脚进行采样。

如果 SDA 为低电平，则已发生了总线冲突 （即另一个主器件正试图发送一个数据 0，见图 19-29）。如果 SDA 被采样为高电平，则 BRG 被重新装入值并开始计数。如果 SDA 在 BRG 超时之前从高电平变为低电平，则不会发生总线冲突，因为两个主器件不可能精确地在同一时刻将 SDA 拉低。

如果 SCL 在 BRG 超时之前从高电平变为低电平，且 SDA 尚未被拉低，那么将发生总线冲突。在此情况下，在重复启动条件期间另一个主器件正试图发送一个数据 1 （见图 19-30）。

如果在 BRG 计时结束时 SCL 和 SDA 都仍然是高电平，则 SDA 引脚被拉低， BRG 重新装入值并开始计数。在计数结束时，不管 SCL 引脚的状态如何， SCL 引脚都被拉低，重复启动条件完成。

图 19-29： 重复启动条件期间的总线冲突 （情形 1）

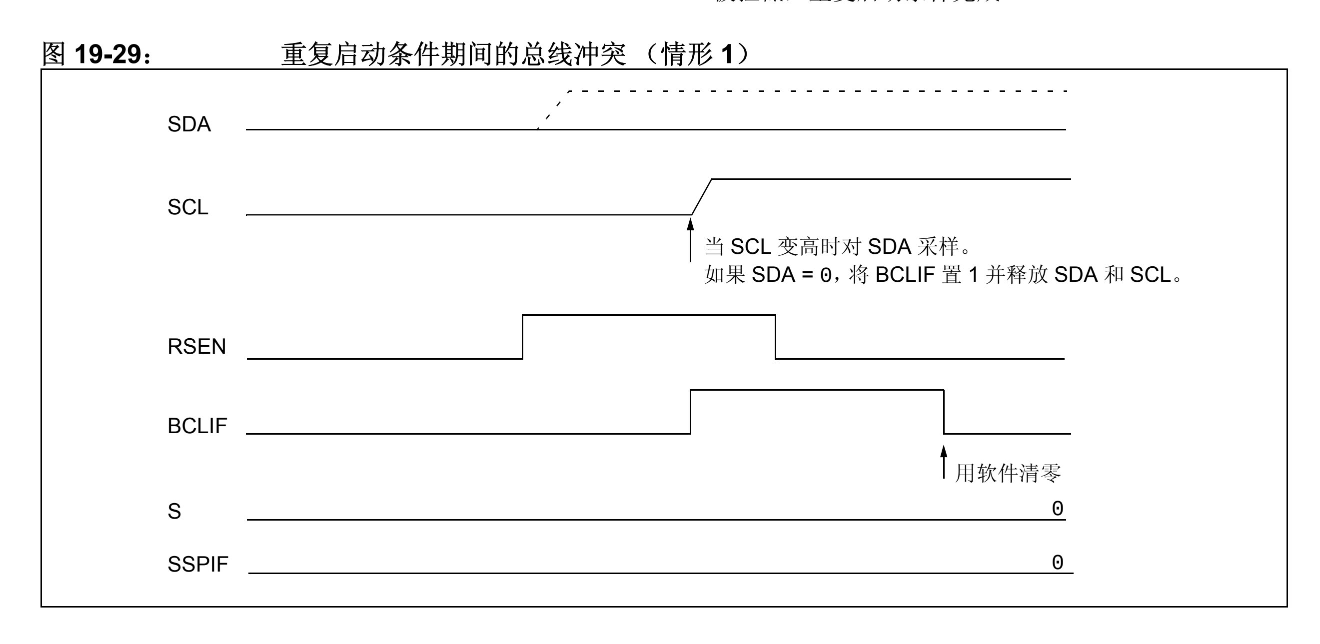
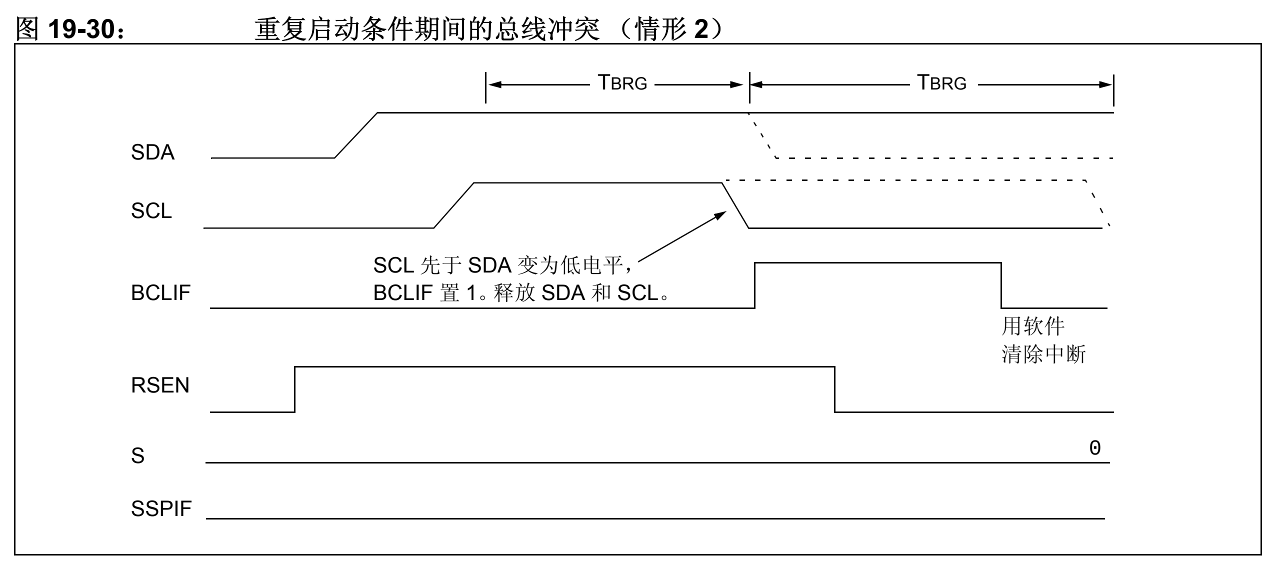


图 19-30： 重复启动条件期间的总线冲突 （情形 2）



 初稿

19.4.17.3 停止条件期间的总线冲突

以下事件会导致停止条件期间的总线冲突：

a) SDA 已被拉高并允许悬空为高电平，但在 BRG 超时后 SDA 被采样为低电平。

b) SCL 引脚被拉高之后，在 SDA 变成高电平之前却被采样为低电平。

停止条件从 SDA 被拉低开始。当采样到 SDA 为低电平时，允许 SCL 引脚悬空。当引脚被采样为高电平时（时钟仲裁），波特率发生器被载入 SSPADD<6:0> 的值并递减计数到 0。当计数器超时后，如果采样到 SDA 为低电平，则表示已发生总线冲突。这是因为另一个主器件正试图发送数据 0（图 19-31）。如果 SCL 引脚在允许 SDA 悬空为高电平前被采样为低电平，也会发生总线冲突。这是上述原因导致的另一种情况 （图 19-32）。

图 **19-31**： 停止条件期间的总线冲突 （情形 **1**）

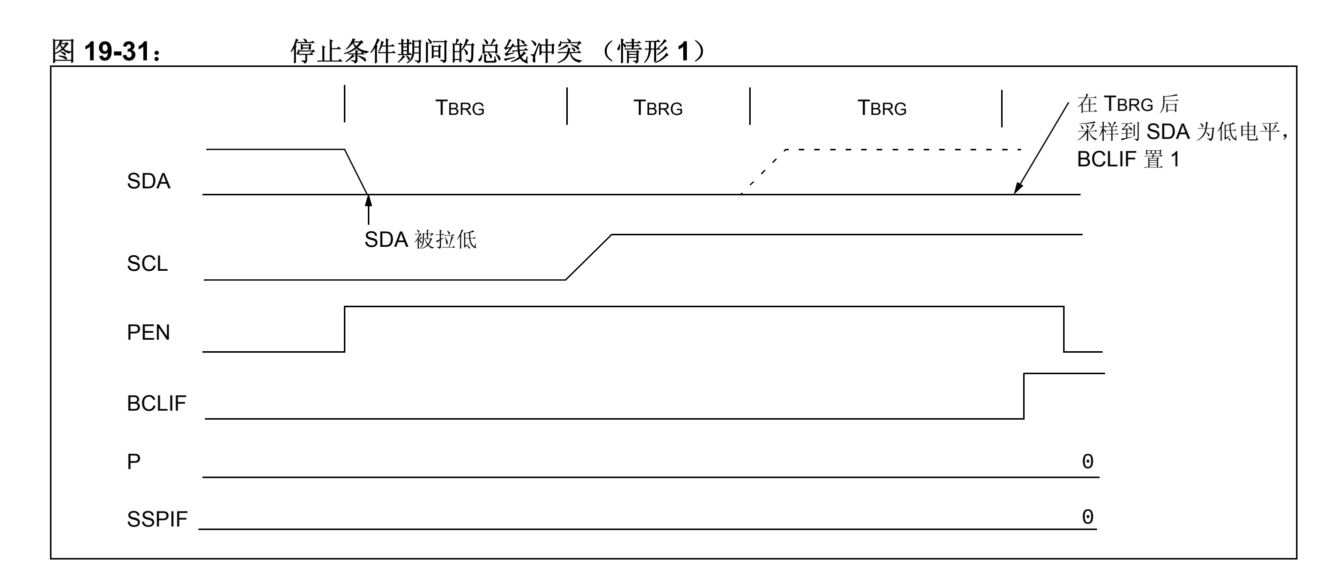


图 **19-32**： 停止条件期间的总线冲突 （情形 **2**）

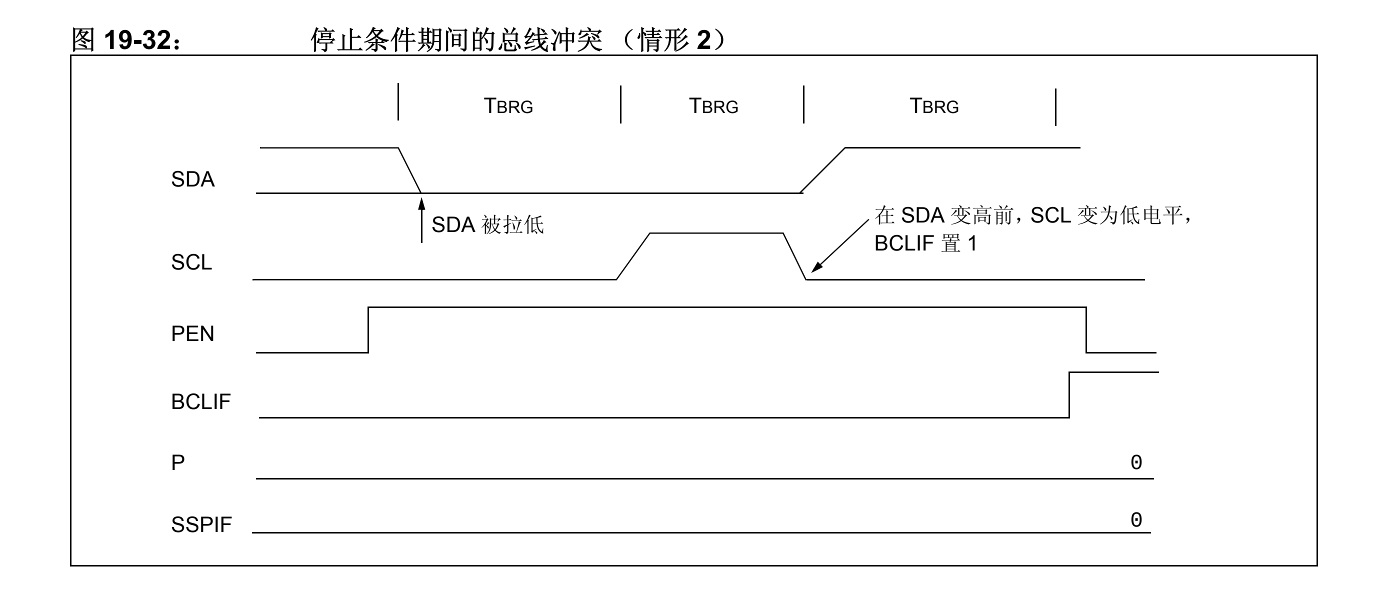


表 19-4： 与 I²C™ 操作相关的寄存器

名称	Bit 7	Bit 6	Bit 5	Bit 4	Bit 3	Bit 2	Bit 1	Bit 0	复位值所在页
INTCON	GIE/GIEH	PEIE/GIEL	TMR0IE	INT0IE	RBIE	TMR0IF	INT0IF	RBIF	51
PIR1	SPPIF(1)	ADIF	RCIF	TXIF	SSPIF	CCP1IF	TMR2IF	TMR1IF	54
PIE1	SPPIE(1)	ADIE	RCIE	TXIE	SSPIE	CCP1IE	TMR2IE	TMR1IE	54
IPR1	SPPIP(1)	ADIP	RCIP	TXIP	SSPIP	CCP1IP	TMR2IP	TMR1IP	54
PIR2	OSCFIF	CMIF	USBIF	EEIF	BCLIF	HLVDIF	TMR3IF	CCP2IF	54
PIE2	OSCFIE	CMIE	USBIE	EEIE	BCLIE	HLVDIE	TMR3IE	CCP2IE	54
IPR2	OSCFIP	CMIP	USBIP	EEIP	BCLIP	HLVDIP	TMR3IP	CCP2IP	54
TRISC	TRISC7	TRISC6	—	—	—	TRISC2	TRISC1	TRISC0	54
TRISD(1)	TRISD7	TRISD6	TRISD5	TRISD4	TRISD3	TRISD2	TRISD1	TRISD0	54
SSPBUF	同步串口接收缓冲器 / 发送寄存器								52
SSPADD	I²C 从动模式下的同步串口地址寄存器 。 I²C 主控模式下的 SSP 波特率寄存器。								52
TMR2	Timer2 寄存器								52
PR2	Timer2 周期寄存器								52
SSPCON1	WCOL	SSPOV	SSPEN	CKP	SSPM3	SSPM2	SSPM1	SSPM0	52
SSPCON2	GCEN	ACKSTAT	ACKDT	ACKEN	RCEN	PEN	RSEN	SEN	52
SSPSTAT	SMP	CKE	D/$\overline{A}$	P	S	R/$\overline{W}$	UA	BF	52

图注： — = 未用 （读为 0）。 I²C™ 模式下的 MSSP 模块不使用阴影单元。

注 1： 28 引脚器件没有使用这些寄存器或位。

 初稿

20.0 增强型通用同步/异步收发器（EUSART）

通用同步 / 异步收发器（Universal Synchronous Asynchronous Receiver Transmitter，USART）模块是两个串行 I/O 模块之一。（USART 也被称为串行通信接口或 SCI。）可以将 USART 配置为能与 CRT 终端和个人计算机等外设通信的全双工异步系统，也可以将它配置成能够与 A/D 或 D/A 集成电路、串行 EEPROM 等外设通信的半双工同步系统。

增强型通用同步 / 异步收发器（EUSART）模块实现了更多的功能，包括自动波特率检测（Automatic Baud Rate Detection，ABD）和校准，以及在接收到“同步间隔”字符和发送 12 位间隔字符时自动唤醒器件。这些功能使 EUSART 模块成为局域互连网络（Local Interconnect Network，LIN）总线系统非常理想的选择。

EUSART 可配置为以下几种工作模式：

- 带有以下功能的全双工异步模式：
  - 字符接收自动唤醒
  - 自动波特率校准
  - 12 位间隔字符发送
- 时钟极性可选的半双工同步主控模式
- 时钟极性可选的半双工同步从动模式

增强型 USART 的引脚与 PORTC 复用。为了将 RC6/TX/CK 和 RC7/RX/DT/SDO 配置为 USART：

- SPEN（RCSTA<7>）位必须置 1（= 1）
- TRISC<7> 位必须置 1（= 1）
- 对于同步 / 异步主控模式，TRISC<6> 必须清零（= 0）；对于同步从动模式，该位必须置 1（= 1）。

注：	EUSART 控制在需要时会自动将引脚从输入重新配置为输出。

增强型 USART 模块的操作由以下 3 个寄存器控制：

- 发送状态和控制寄存器（TXSTA）
- 接收状态和控制寄存器（RCSTA）
- 波特率控制寄存器（BAUDCON）

在后面几页中的寄存器 20-1、寄存器 20-2 和寄存器 20-3 分别对这些寄存器作出了详细说明。

寄存器 20-1： TXSTA：发送状态和控制寄存器

R/W-0	R/W-0	R/W-0	R/W-0	R/W-0	R/W-0	R-1	R/W-0
CSRC	TX9	TXEN	SYNC	SENDB	BRGH	TRMT	TX9D
bit 7							bit 0

bit 7 **CSRC**：时钟源选择位

异步模式：
忽略。

同步模式：
1 = 主控模式 （时钟来自内部 BRG）
0 = 从动模式 （时钟来自外部时钟源）

bit 6 **TX9**：9 位发送使能位
1 = 选择 9 位发送
0 = 选择 8 位发送

bit 5 **TXEN**：发送使能位
1 = 使能发送
0 = 禁止发送

注： 同步模式下 SREN/CREN 的优先级高于 TXEN，但同步从动模式除外，SREN 位在该模式下不起作用。

bit 4 **SYNC**：EUSART 模式选择位
1 = 同步模式
0 = 异步模式

bit 3 **SENDB**：发送间隔字符位

异步模式：
1 = 在下一次发送时发送 “同步间隔”字符 （在完成时由硬件清零）
0 = “同步间隔”字符发送完成

同步模式：
忽略。

bit 2 **BRGH**：高波特率选择位

异步模式：
1 = 高速
0 = 低速

同步模式：
在此模式下未使用。

bit 1 **TRMT**：发送移位寄存器状态位
1 = TSR 空
0 = TSR 满

bit 0 **TX9D**：发送数据的第 9 位
该位可以是地址 / 数据位或奇偶校验位。

图注：			
R = 可读位	W = 可写位	U = 未用位，读为 0	
-n = 上电复位时的值	1 = 置 1	0 = 清零	x = 未知

 初稿

寄存器 20-2： RCSTA：接收状态和控制寄存器

R/W-0	R/W-0	R/W-0	R/W-0	R/W-0	R-0	R-0	R-x
SPEN	RX9	SREN	CREN	ADDEN	FERR	OERR	RX9D
bit 7							bit 0

bit 7 **SPEN**：串口使能位
1 = 使能串口 （将 RX/DT 和 TX/CK 引脚配置为串口引脚）
0 = 禁止串口 （保持在复位状态）

bit 6 **RX9**：9 位接收使能位
1 = 选择 9 位接收
0 = 选择 8 位接收

bit 5 **SREN**：单字节接收使能位
异步模式：
忽略。
同步主控模式：
1 = 使能单字节接收
0 = 禁止单字节接收
此位在接收完成后清零。
同步从动模式：
忽略。

bit 4 **CREN**：连续接收使能位
异步模式：
1 = 使能接收器
0 = 禁止接收器
同步模式：
1 = 使能连续接收，直到使能位 CREN 清零 （CREN 的优先级比 SREN 高）
0 = 禁止连续接收

bit 3 **ADDEN**：地址检测使能位
9 位异步模式 （RX9 = 1）：
1 = 当 RSR<8> 置 1 时，使能地址检测、允许中断并装载接收缓冲器
0 = 禁止地址检测，接收到所有字节并且第 9 位可用作奇偶校验位
9 位异步模式 （RX9 = 0）：
忽略。

bit 2 **FERR**：帧错误位
1 = 帧错误 （可以通过读 RCREG 寄存器刷新并接收下一个有效字节）
0 = 无帧错误

bit 1 **OERR**：溢出错误位
1 = 溢出错误 （可以通过清零 CREN 位清零）
0 = 无溢出错误

bit 0 **RX9D**：接收数据的第 9 位
该位可以是地址 / 数据位或奇偶校验位，必须由用户固件计算得到。

图注：			
R = 可读位	W = 可写位	U = 未用位，读为 0	
-n = 上电复位时的值	1 = 置 1	0 = 清零	x = 未知

寄存器 20-3： **BAUDCON**：波特率控制寄存器

R/W-0	R-1	U-0	R/W-0	R/W-0	U-0	R/W-0	R/W-0
ABDOVF	RCIDL	—	SCKP	BRG16	—	WUE	ABDEN
bit 7							bit 0

bit 7 **ABDOVF**：自动波特率采样计满返回状态位
1 = 在自动波特率检测模式下发生了 BRG 计满返回 （必须由软件清零）
0 = 没有发生 BRG 计满返回

bit 6 **RCIDL**：接收操作空闲状态位
1 = 接收操作处于空闲状态
0 = 接收操作处于活动状态

bit 5 未用：读为 0

bit 4 **SCKP**：同步时钟极性选择位
异步模式：
在此模式下未使用。
同步模式：
1= 空闲状态时钟 （CK）为高电平
0= 空闲状态时钟 （CK）为低电平

bit 3 **BRG16**：16 位波特率寄存器使能位
1 = 16 位波特率发生器——SPBRGH 和 SPBRG
0 = 8 位波特率发生器——仅 SPBRG （兼容模式），忽略 SPBRGH 的值

bit 2 未用：读为 0

bit 1 **WUE**：唤醒使能位
异步模式：
1 = EUSART 将继续采样 RX 引脚——中断在下降沿产生，在下一个上升沿由硬件清零该位
0 = 未监视 RX 引脚或检测到了上升沿
同步模式：
在此模式下未使用。

bit 0 **ABDEN**：自动波特率检测使能位
异步模式：
1 = 在下一个字符使能波特率检测。需要收到 “同步”字段 （55h）。完成时由硬件清零。
0 = 禁止波特率检测或检测已完成
同步模式：
在此模式下未使用。

图注：			
R = 可读位	W = 可写位	U = 未用位，读为 0	
-n = 上电复位时的值	1 = 置 1	0 = 清零	x = 未知

20.1 波特率发生器 （BRG）

BRG 是一个专用的 8 位或 16 位发生器，支持 EUSART 的异步和同步模式。默认情况下，BRG 工作在 8 位模式下，将 BRG16 （BAUDCON<3>）位置 1 可选择 16 位模式。

SPBRGH:SPBRG 寄存器对控制独立运行定时器的周期。在异步模式下，BRGH（TXSTA<2>）位和 BRG16（BAUDCON<3>）位也控制波特率。在同步模式下，会忽略 BRGH 位。表 20-1 所示为不同 EUSART 模式的波特率计算公式，但仅适用于主控模式（由内部产生时钟信号）。

给定目标波特率和 Fosc 的情况下，可以使用表 20-1 中的公式计算 SPBRGH:SPBRG 寄存器的最近似整数值，从而确定波特率误差。例 20-1 给出了一个计算示例。表 20-2 中给出了各种异步模式下典型的波特率和误差值。使用高波特率 （BRGH = 1）或 16 位 BRG 有利于减小波特率误差，或者在快速振荡频率条件下实现低波特率。

向 SPBRGH:SPBRG 寄存器写入新值会导致 BRG 定时器复位 （或清零）。这可以确保 BRG 无需等待定时器溢出就可以输出新的波特率。

20.1.1 在功耗管理模式下的操作

器件时钟用于产生所需的波特率。当进入一种功耗管理模式时，新时钟源可能会工作在与先前不同的频率下。这可能需要调整 SPBRG 寄存器对中的值。

20.1.2 采样

择多检测电路对 RX 引脚采样三次，以判定 RX 引脚上出现的是高电平还是低电平。

表 20-1： 波特率公式

配置位			BRG/EUSART 模式	波特率公式
SYNC	BRG16	BRGH		
0	0	0	8 位 / 异步	Fosc/[64 (n + 1)]
0	0	1	8 位 / 异步	Fosc/[16 (n + 1)]
0	1	0	16 位 / 异步	
0	1	1	16 位 / 异步	Fosc/[4 (n + 1)]
1	0	x	8 位 / 同步	
1	1	x	16 位 / 同步	

图注： x = 任意值， n = SPBRGH:SPBRG 寄存器对的值

例 20-1： 计算波特率误差

针对工作在异步模式下，工作频率 Fosc 为 16 MHz，目标波特率为 9600bps 的 8 位 BRG：

目标波特率 = Fosc/(64 ([SPBRGH:SPBRG] + 1))

求解 SPBRGH:SPBRG：

X = ((Fosc/ 目标波特率)/64) – 1
= ((16000000/9600)/64) – 1
= [25.042] = 25

计算得到的波特率 = 16000000/(64 (25 + 1))
= 9615

误差 = (波特率计算结果 – 目标波特率)/ 目标波特率
= (9615 - 9600)/9600 = 0.16%

表 20-2： 与波特率发生器相关的寄存器

名称	Bit 7	Bit 6	Bit 5	Bit 4	Bit 3	Bit 2	Bit 1	Bit 0	复位值所在页
TXSTA	CSRC	TX9	TXEN	SYNC	SENDB	BRGH	TRMT	TX9D	53
RCSTA	SPEN	RX9	SREN	CREN	ADDEN	FERR	OERR	RX9D	53
BAUDCON	ABDOVF	RCIDL	—	SCKP	BRG16	—	WUE	ABDEN	53
SPBRGH	EUSART 波特率发生器寄存器的高字节								53
SPBRG	EUSART 波特率发生器寄存器的低字节								53

图注： — = 未用 （读为 0）。 BRG 未使用阴影单元。

表 20-3： 异步模式的波特率

目标波特率（Kbps）	SYNC = 0， BRGH = 0， BRG16 = 0											
	Fosc = 40.000 MHz			Fosc = 20.000 MHz			Fosc = 10.000 MHz			Fosc = 8.000 MHz		
	实际波特率（Kbps）	% 误差	SPBRG 值（10 进制）	实际波特率（Kbps）	% 误差	SPBRG 值（10 进制）	实际波特率（Kbps）	% 误差	SPBRG 值（10 进制）	实际波特率（Kbps）	% 误差	SPBRG 值（10 进制）
0.3	—	—	—	—	—	—	—	—	—	—	—	—
1.2	—	—	—	1.221	1.73	255	1.202	0.16	129	1.201	-0.16	103
2.4	2.441	1.73	255	2.404	0.16	129	2.404	0.16	64	2.403	-0.16	51
9.6	9.615	0.16	64	9.766	1.73	31	9.766	1.73	15	9.615	-0.16	12
19.2	19.531	1.73	31	19.531	1.73	15	19.531	1.73	7	—	—	—
57.6	56.818	-1.36	10	62.500	8.51	4	52.083	-9.58	2	—	—	—
115.2	125.000	8.51	4	104.167	-9.58	2	78.125	-32.18	1	—	—	—

目标波特率（Kbps）	SYNC = 0， BRGH = 0， BRG16 = 0								
	Fosc = 4.000 MHz			Fosc = 2.000 MHz			Fosc = 1.000 MHz		
	实际波特率（Kbps）	% 误差	SPBRG 值（10 进制）	实际波特率（Kbps）	% 误差	SPBRG 值（10 进制）	实际波特率（Kbps）	% 误差	SPBRG 值（10 进制）
0.3	0.300	0.16	207	0.300	-0.16	103	0.300	-0.16	51
1.2	1.202	0.16	51	1.201	-0.16	25	1.201	-0.16	12
2.4	2.404	0.16	25	2.403	-0.16	12	—	—	—
9.6	8.929	-6.99	6	—	—	—	—	—	—
19.2	20.833	8.51	2	—	—	—	—	—	—
57.6	62.500	8.51	0	—	—	—	—	—	—
115.2	62.500	-45.75	0	—	—	—	—	—	—

目标波特率（Kbps）	SYNC = 0， BRGH = 1， BRG16 = 0											
	Fosc = 40.000 MHz			Fosc = 20.000 MHz			Fosc = 10.000 MHz			Fosc = 8.000 MHz		
	实际波特率（Kbps）	% 误差	SPBRG 值（10 进制）	实际波特率（Kbps）	% 误差	SPBRG 值（10 进制）	实际波特率（Kbps）	% 误差	SPBRG 值（10 进制）	实际波特率（Kbps）	% 误差	SPBRG 值（10 进制）
0.3	—	—	—	—	—	—	—	—	—	—	—	—
1.2	—	—	—	—	—	—	—	—	—	—	—	—
2.4	—	—	—	—	—	—	2.441	1.73	255	2.403	-0.16	207
9.6	9.766	1.73	255	9.615	0.16	129	9.615	0.16	64	9.615	-0.16	51
19.2	19.231	0.16	129	19.231	0.16	64	19.531	1.73	31	19.230	-0.16	25
57.6	58.140	0.94	42	56.818	-1.36	21	56.818	-1.36	10	55.555	3.55	8
115.2	113.636	-1.36	21	113.636	-1.36	10	125.000	8.51	4	—	—	—

目标波特率（Kbps）	SYNC = 0， BRGH = 1， BRG16 = 0								
	Fosc = 4.000 MHz			Fosc = 2.000 MHz			Fosc = 1.000 MHz		
	实际波特率（Kbps）	% 误差	SPBRG 值（10 进制）	实际波特率（Kbps）	% 误差	SPBRG 值（10 进制）	实际波特率（Kbps）	% 误差	SPBRG 值（10 进制）
0.3	—	—	—	—	—	—	0.300	-0.16	207
1.2	1.202	0.16	207	1.201	-0.16	103	1.201	-0.16	51
2.4	2.404	0.16	103	2.403	-0.16	51	2.403	-0.16	25
9.6	9.615	0.16	25	9.615	-0.16	12	—	—	—
19.2	19.231	0.16	12	—	—	—	—	—	—
57.6	62.500	8.51	3	—	—	—	—	—	—
115.2	125.000	8.51	1	—	—	—	—	—	—

表 **20-3**： 异步模式的波特率 （续）

目标波特率（Kbps）	SYNC = 0， BRGH = 0， BRG16 = 1											
	Fosc = 40.000 MHz			Fosc = 20.000 MHz			Fosc = 10.000 MHz			Fosc = 8.000 MHz		
	实际波特率（Kbps）	% 误差	SPBRG 值（10 进制）	实际波特率（Kbps）	% 误差	SPBRG 值（10 进制）	实际波特率（Kbps）	% 误差	SPBRG 值（10 进制）	实际波特率（Kbps）	% 误差	SPBRG 值（10 进制）
0.3	0.300	0.00	8332	0.300	0.02	4165	0.300	0.02	2082	0.300	-0.04	1665
1.2	1.200	0.02	2082	1.200	-0.03	1041	1.200	-0.03	520	1.201	-0.16	415
2.4	2.402	0.06	1040	2.399	-0.03	520	2.404	0.16	259	2.403	-0.16	207
9.6	9.615	0.16	259	9.615	0.16	129	9.615	0.16	64	9.615	-0.16	51
19.2	19.231	0.16	129	19.231	0.16	64	19.531	1.73	31	19.230	-0.16	25
57.6	58.140	0.94	42	56.818	-1.36	21	56.818	-1.36	10	55.555	3.55	8
115.2	113.636	-1.36	21	113.636	-1.36	10	125.000	8.51	4	—	—	—

目标波特率（Kbps）	SYNC = 0， BRGH = 0， BRG16 = 1								
	Fosc = 4.000 MHz			Fosc = 2.000 MHz			Fosc = 1.000 MHz		
	实际波特率（Kbps）	% 误差	SPBRG 值（10 进制）	实际波特率（Kbps）	% 误差	SPBRG 值（10 进制）	实际波特率（Kbps）	% 误差	SPBRG 值（10 进制）
0.3	0.300	0.04	832	0.300	-0.16	415	0.300	-0.16	207
1.2	1.202	0.16	207	1.201	-0.16	103	1.201	-0.16	51
2.4	2.404	0.16	103	2.403	-0.16	51	2.403	-0.16	25
9.6	9.615	0.16	25	9.615	-0.16	12	—	—	—
19.2	19.231	0.16	12	—	—	—	—	—	—
57.6	62.500	8.51	3	—	—	—	—	—	—
115.2	125.000	8.51	1	—	—	—	—	—	—

目标波特率（Kbps）	SYNC = 0, BRGH = 1, BRG16 = 1 或 SYNC = 1, BRG16 = 1											
	Fosc = 40.000 MHz			Fosc = 20.000 MHz			Fosc = 10.000 MHz			Fosc = 8.000 MHz		
	实际波特率（Kbps）	% 误差	SPBRG 值（10 进制）	实际波特率（Kbps）	% 误差	SPBRG 值（10 进制）	实际波特率（Kbps）	% 误差	SPBRG 值（10 进制）	实际波特率（Kbps）	% 误差	SPBRG 值（10 进制）
0.3	0.300	0.00	33332	0.300	0.00	16665	0.300	0.00	8332	0.300	-0.01	6665
1.2	1.200	0.00	8332	1.200	0.02	4165	1.200	0.02	2082	1.200	-0.04	1665
2.4	2.400	0.02	4165	2.400	0.02	2082	2.402	0.06	1040	2.400	-0.04	832
9.6	9.606	0.06	1040	9.596	-0.03	520	9.615	0.16	259	9.615	-0.16	207
19.2	19.193	-0.03	520	19.231	0.16	259	19.231	0.16	129	19.230	-0.16	103
57.6	57.803	0.35	172	57.471	-0.22	86	58.140	0.94	42	57.142	0.79	34
115.2	114.943	-0.22	86	116.279	0.94	42	113.636	-1.36	21	117.647	-2.12	16

目标波特率（Kbps）	SYNC = 0, BRGH = 1, BRG16 = 1 或 SYNC = 1, BRG16 = 1								
	Fosc = 4.000 MHz			Fosc = 2.000 MHz			Fosc = 1.000 MHz		
	实际波特率（Kbps）	% 误差	SPBRG 值（10 进制）	实际波特率（Kbps）	% 误差	SPBRG 值（10 进制）	实际波特率（Kbps）	% 误差	SPBRG 值（10 进制）
0.3	0.300	0.01	3332	0.300	-0.04	1665	0.300	-0.04	832
1.2	1.200	0.04	832	1.201	-0.16	415	1.201	-0.16	207
2.4	2.404	0.16	415	2.403	-0.16	207	2.403	-0.16	103
9.6	9.615	0.16	103	9.615	-0.16	51	9.615	-0.16	25
19.2	19.231	0.16	51	19.230	-0.16	25	19.230	-0.16	12
57.6	58.824	2.12	16	55.555	3.55	8	—	—	—
115.2	111.111	-3.55	8	—	—	—	—	—	—

20.1.3 自动波特率检测

增强型 USART 模块支持波特率自动检测和校准。此功能仅在异步模式下当 WUE 位清零时有效。

只要接收到起始位并且 ABDEN 位已置 1，就会开始自动波特率检测（图 20-1）。波特率计算采用自平均的方式。

在自动波特率检测（ABD）模式下，BRG 的时钟是反向的。不是由 BRG 为 RX 输入信号提供时钟源，而是由 RX 信号为 BRG 定时。在 ABD 模式下，内部波特率发生器被用作计数器，来计算输入的串行字节流的位间隔时间。

一旦 ABDEN 位置 1，状态机就会将 BRG 清零并寻找起始位。为了正确计算比特率，自动波特率检测必须接收到一个值为 55h（ASCII 字符 U，也是 LIN 总线的同步字符）的字节。为了尽量减少输入信号不对称所造成的影响，在接收低位和高位的时间内都要进行测量。在起始位后，SPBRG 使用预先选择的时钟源，在 RX 引脚信号的第一个上升沿开始计数。在 RX 引脚传输了 8 个位，或在检测到第 5 个上升沿后，会将在相应 BRG 周期内累计的值保存在SPBRGH:SPBRG寄存器对中。当第 5 个时钟周期出现时（应与停止位对应），ABDEN 位会自动清零。

如果发生了 BRG 计满返回（从 FFFFh 到 0000h 的溢出），会在 ABDOVF 状态位（BAUDCON<7>）有所反映。当 BRG 计满返回时，该位由硬件置 1，用户也可用软件将其置 1 或清零。在发生计满返回事件后，继续保持 ABD 模式，ABDEN 位保持置 1（图 20-2）。

当校准波特率周期时，BRG寄存器时钟频率为预配置时钟频率的 1/8。请注意 BRG 时钟由 BRG16 和 BRGH 位配置。无论BRG16位的设置如何，SPBRG和SPBRGH都将被用作一个 16 位计数器。通过检查 SPBRGH 寄存器中的值是否为 00h，用户可以验证在 8 位模式下是否发生了进位。表 20-4 所示为 BRG 计数器的时钟速率。

当产生ABD序列时，EUSART状态机保持在空闲状态。一旦在 RX 上检测到第 5 个上升沿，中断标志位 RCIF 就会置 1。需要读取 RCREG 中的值，来清除中断标志位 RCIF。应丢弃 RCREG 的内容。

> **注 1：** 如果 WUE 位与 ABDEN 位同时置 1，自动波特率检测会在间隔字符*之后*的字节开始。
>
> **2：** 需要用户来判断进入的字符波特率是否处于所选的 BRG 时钟源范围内。由于位误差率的原因，某些振荡频率和 EUSART 波特率的组合是无法实现的。在使用自动波特率检测功能时，必须综合考虑系统总的时序和通信波特率。

表 20-4： BRG 计数器时钟速率

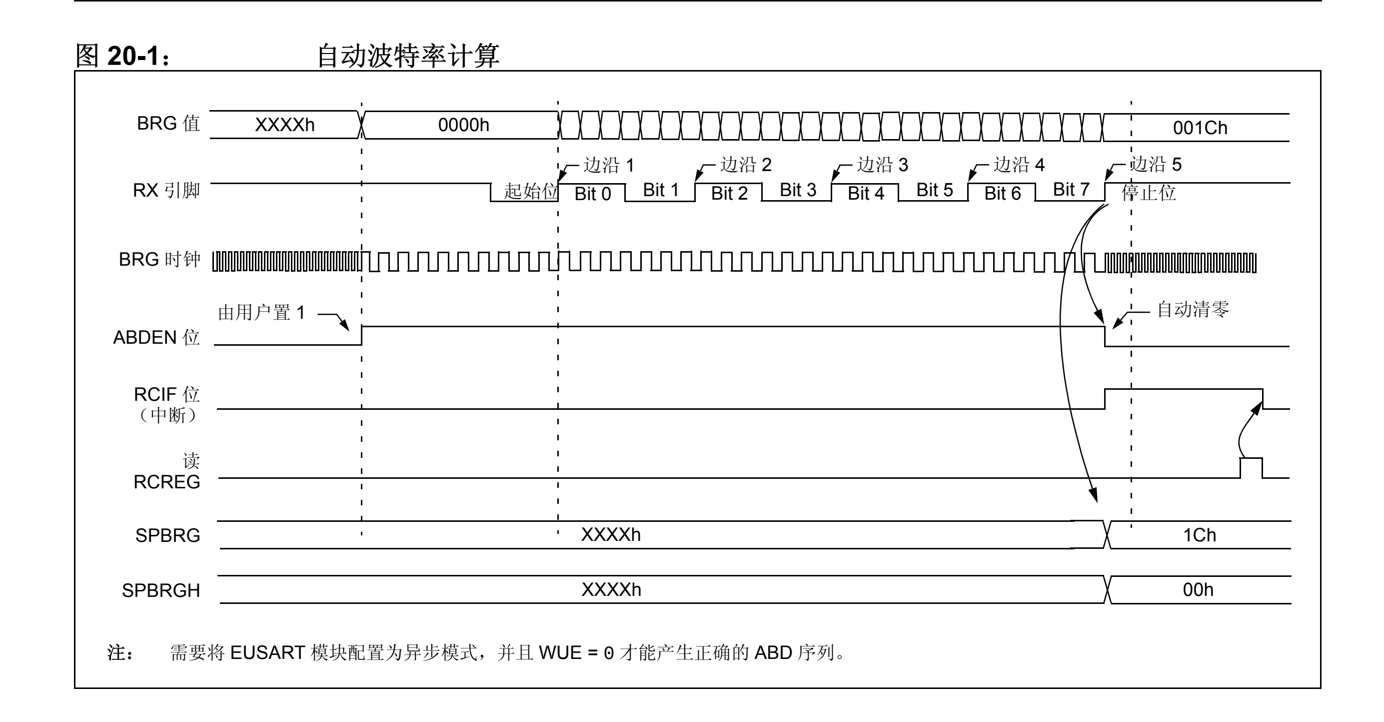
BRG16	BRGH	BRG 计数器时钟
0	0	FOSC/512
0	1	FOSC/128
1	0	FOSC/128
1	1	FOSC/32

注： 在产生 ABD 序列时，不管 BRG16 的设置如何，SPBRG 和 SPBRGH 被用作一个 16 位计数器。

20.1.3.1 ABD 和 EUSART 发送

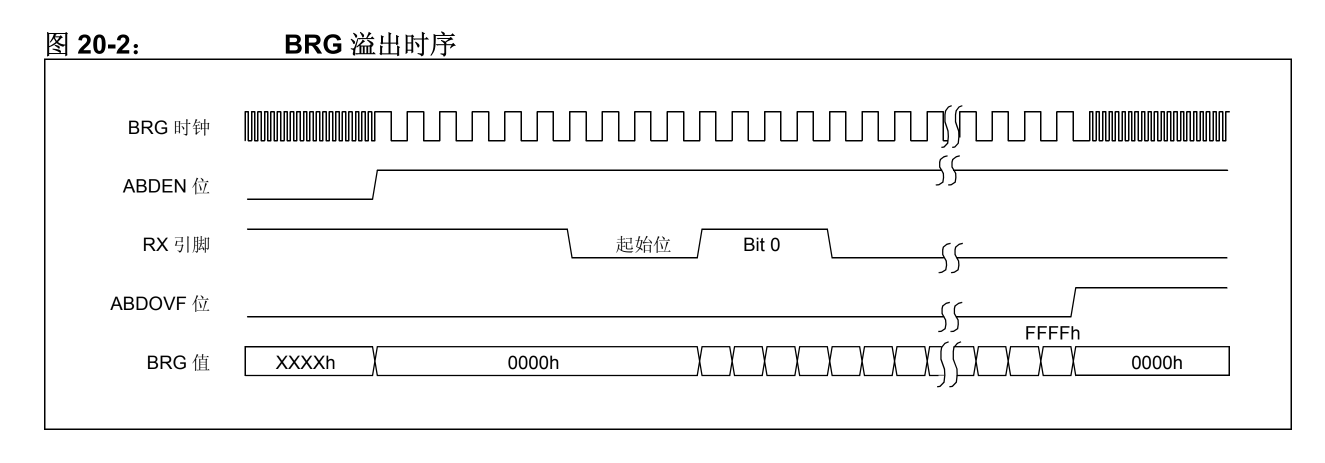
由于在ABD采集期间BRG时钟是反向的，因此在ABD期间不能使用 EUSART 发送器。这意味着只要 ABDEN 位置 1，就不能写入 TXREG。用户还应确保在发送期间 ABDEN 不会是置 1 状态，否则可能会导致无法预料的 EUSART 操作。

图 **20-1：** 自动波特率计算



注： 需要将 EUSART 模块配置为异步模式，并且 WUE = 0 才能产生正确的 ABD 序列。

图 **20-2：** **BRG** 溢出时序



20.2 EUSART 异步模式

通过将 SYNC 位（TXSTA<4>）清零可选择异步工作模式。在此模式下， EUSART 使用标准的不归零（Non-Return-to-Zero，NRZ）格式（1 个起始位，8 个或 9 个数据位和 1 个停止位）。最常用的格式为 8 位数据。片上专用 8 位 /16 位波特率发生器可借助于振荡器产生标准波特率频率。

EUSART 首先发送和接收的是最低有效位。EUSART 的发送器和接收器在功能上是独立的，但采用相同的数据格式和波特率。波特率发生器可以根据 BRGH 和 BRG16 位（TXSTA<2> 和 BAUDCON<3>）的设置值产生两种不同的波特率时钟，频率分别为移位速率的 16 倍和 64 倍。硬件不支持奇偶校验，但可以用软件实现，校验位保存在第 9 个数据位。

当工作在异步模式下时，EUSART 模块包括以下重要组成部分：

- 波特率发生器
- 采样电路
- 异步发送器
- 异步接收器
- 同步间隔字符自动唤醒
- 12 位间隔字符发送
- 自动波特率检测

20.2.1 EUSART 异步发送器

图 20-3 显示了 EUSART 发送器的框图。发送器的核心是发送（串行）移位寄存器（Transmit Shift Register，TSR）。移位寄存器从读 / 写发送缓冲寄存器 TXREG 中获取数据。 TXREG 寄存器中的数据由软件装入。直到前一次装入的停止位已被发送，才会向 TSR 寄存器装入新数据。一旦停止位发送完毕， TXREG 寄存器中的新数据（如果有的话）就会被装入 TSR。

一旦 TXREG 寄存器向 TSR 寄存器传输了数据（在 1 个 TCY 内发生），TXREG 寄存器就为空，同时标志位 TXIF（PIR1<4>）置 1。可以通过将中断允许位 TXIE（PIE1<4>）置 1 或清零来允许 / 禁止该中断。不管 TXIE 的状态如何，只要中断发生，TXIF 就会置 1 并且不能用软件清零。TXIF 不会在 TXREG 装入新数据时立即被清零，而是在装入指令后的第二个指令周期复位。因此在 TXREG装入新数据后立即查询TXIF，会返回无效结果。

标志位 TXIF 表示的是 TXREG 寄存器的状态，而另一个位 TRMT（TXSTA<1>）则表示 TSR 寄存器的状态。TRMT是只读位，它在TSR寄存器为空时被置1。TRMT位与任何中断均无关联，因此要确定TSR 寄存器是否为空，用户只能对此位进行轮询。

> 注 1：TSR 寄存器并未映射到数据存储器中，因此用户不能访问它。
>
> 2：当使能位 TXEN 置 1 时，标志位 TXIF 也置 1。

设置异步发送的操作步骤如下：

1. 对SPBRGH:SPBRG寄存器进行初始化，以设置合适的波特率。按需要将 BRGH 和 BRG16 位置 1 或清零，以获得目标波特率。
2. 通过将 SYNC 位清零并将 SPEN 位置 1 使能异步串口。
3. 如果需要中断，将使能位 TXIE 置 1。
4. 若需要发送 9 位数据，请将发送位 TX9 置 1。发送的第 9 位可以是地址位也可以是数据位。
5. 通过将 TXEN 位置 1 使能发送，此操作同时也会将 TXIF 位置 1。
6. 如果选择发送 9 位数据，应该将第 9 位数据装入 TX9D 位。
7. 将数据装入 TXREG 寄存器 （开始发送）。
8. 若想使用中断，请确保将 INTCON 寄存器中的 GIE 和 PEIE 位 （INTCON<7:6>）置 1。

图 20-3： EUSART 发送框图

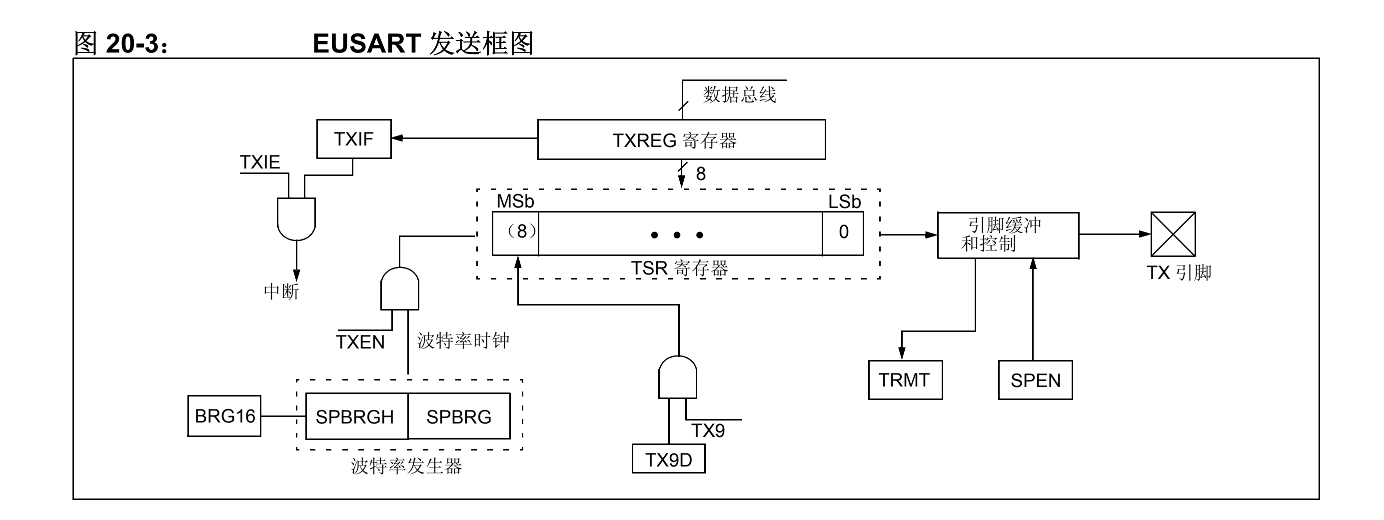


图 20-4：　　异步发送

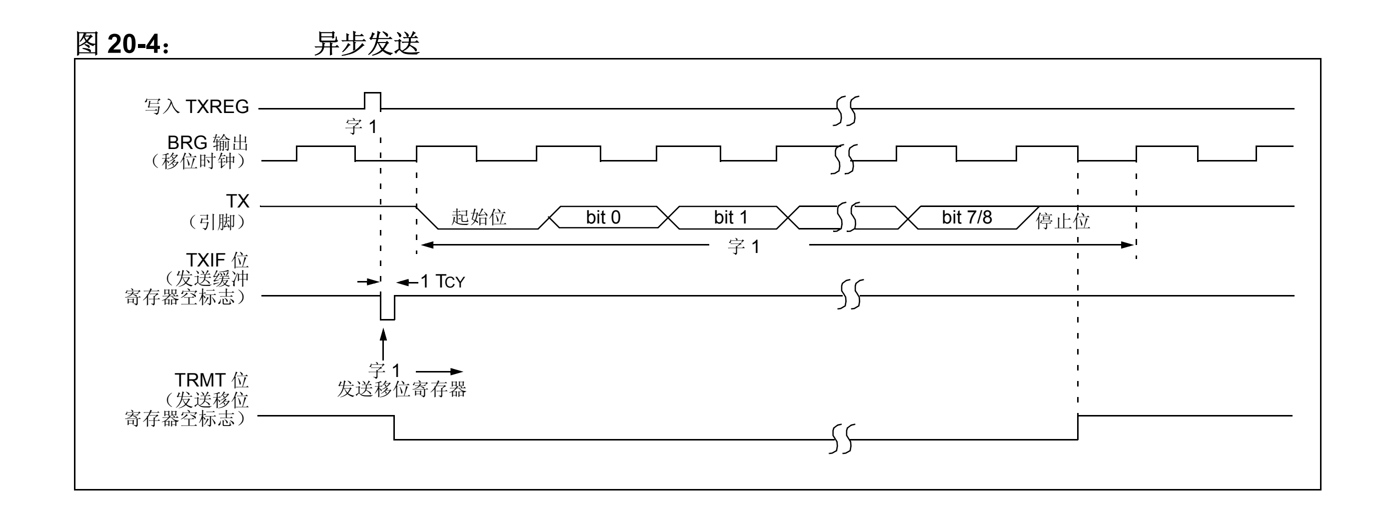
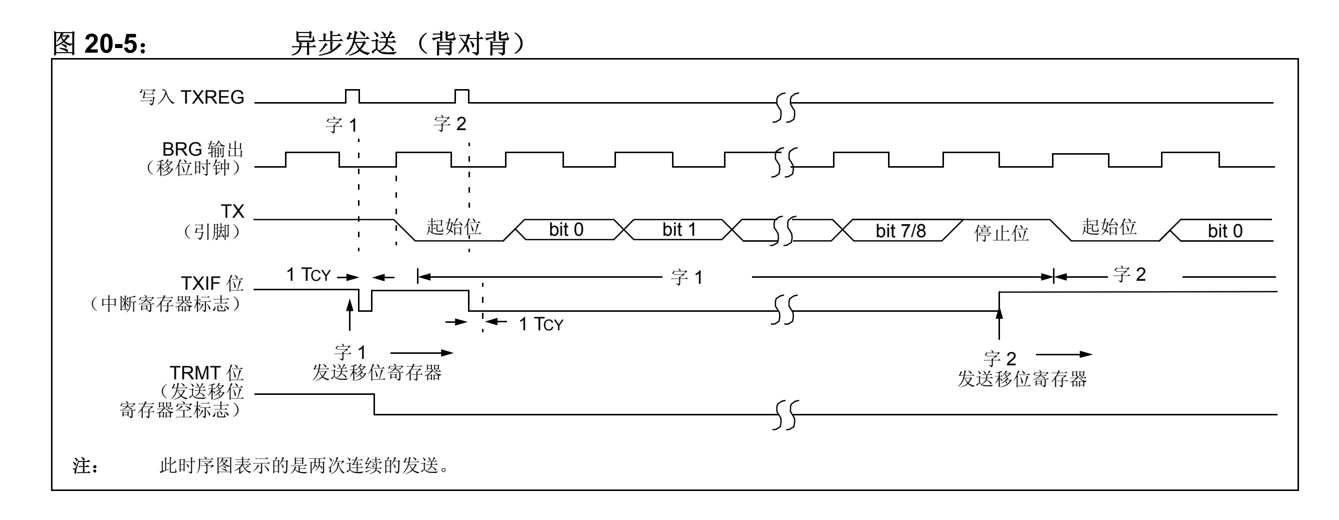


图 20-5：　　异步发送 （背对背）



注：　此时序图表示的是两次连续的发送。

表 20-5：　　与异步发送相关的寄存器

名称	Bit 7	Bit 6	Bit 5	Bit 4	Bit 3	Bit 2	Bit 1	Bit 0	复位值所在页
INTCON	GIE/GIEH	PEIE/GIEL	TMR0IE	INT0IE	RBIE	TMR0IF	INT0IF	RBIF	51
PIR1	SPPIF[1]	ADIF	RCIF	TXIF	SSPIF	CCP1IF	TMR2IF	TMR1IF	54
PIE1	SPPIE[1]	ADIE	RCIE	TXIE	SSPIE	CCP1IE	TMR2IE	TMR1IE	54
IPR1	SPPIP[1]	ADIP	RCIP	TXIP	SSPIP	CCP1IP	TMR2IP	TMR1IP	54
RCSTA	SPEN	RX9	SREN	CREN	ADDEN	FERR	OERR	RX9D	53
TXREG	EUSART 发送寄存器								53
TXSTA	CSRC	TX9	TXEN	SYNC	SENDB	BRGH	TRMT	TX9D	53
BAUDCON	ABDOVF	RCIDL	—	SCKP	BRG16	—	WUE	ABDEN	53
SPBRGH	EUSART 波特率发生器寄存器的高字节								53
SPBRG	EUSART 波特率发生器寄存器的低字节								53

图注：　— = 未用 （读为 0）。异步发送不使用阴影单元。

注　1：　在 28 引脚器件上保留；应始终保持这些位清零。

20.2.2 EUSART 异步接收器

图 20-6 显示了接收器的框图。在 RX 引脚上接收数据，并驱动数据恢复电路。数据恢复电路实际上是一个高速移位器，其工作频率是波特率的 16 倍，而主接收串行移位器的工作频率等于比特率或 FOSC。此模式通常用于 RS-232 系统。

设置异步接收操作的步骤如下：

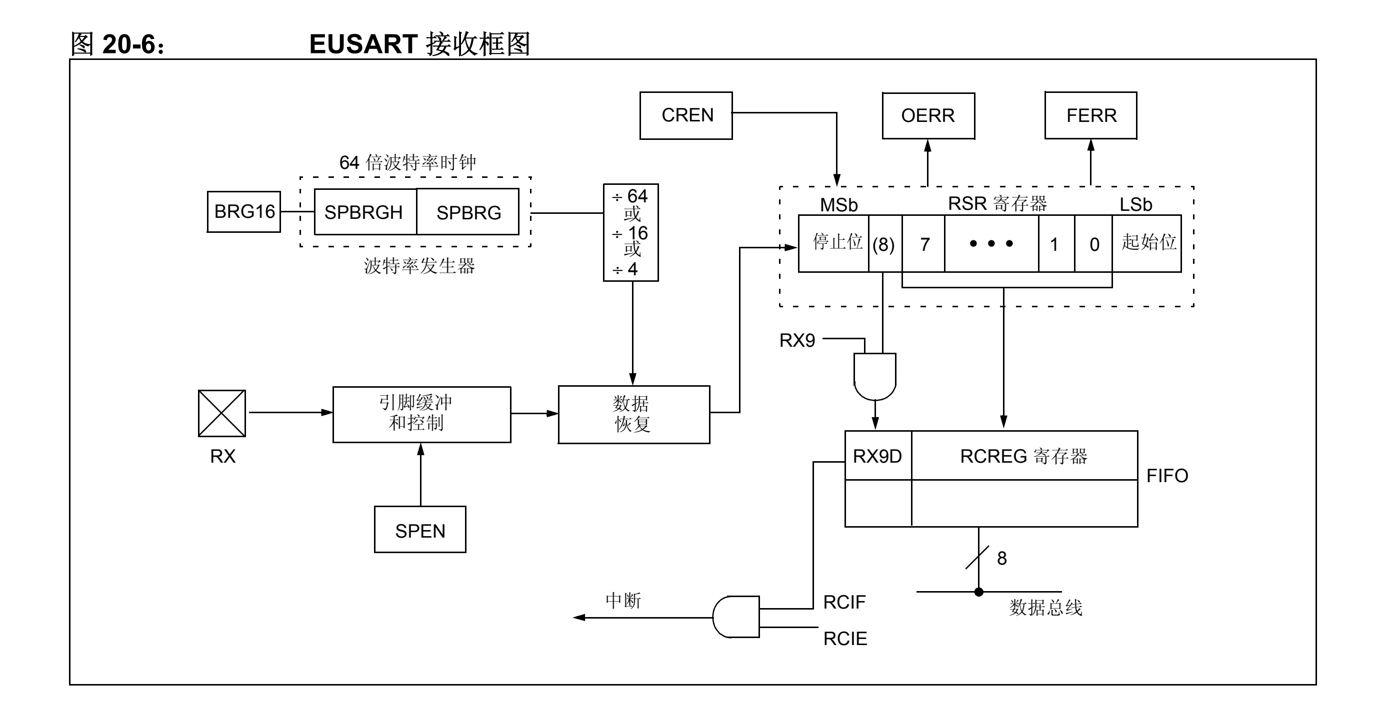
1. 对 SPBRGH:SPBRG 寄存器进行初始化，以设置合适的波特率。按需要将 BRGH 和 BRG16 位置 1 或清零，以获得目标波特率。
2. 通过将 SYNC 位清零并将 SPEN 位置 1 使能异步串口。
3. 如果需要中断，将使能位 RCIE 置 1。
4. 若需要接收 9 位数据，将 RX9 位置 1。
5. 通过将 CREN 位置 1，使能接收。
6. 当接收完成时标志位 RCIF 将置 1，此时如果使能位 RCIE 已置 1，还将产生一个中断。
7. 读 RCSTA 寄存器以获取第 9 位数据（如果已使能），并判断在接收过程中是否发生了错误。
8. 通过读RCREG寄存器来读取接收到的8位数据。
9. 如果发生错误，通过将使能位 CREN 清零来清除错误。
10. 若想使用中断，请确保将 INTCON 寄存器中的 GIE 和 PEIE 位 （INTCON<7:6>）置 1。

20.2.3 设置带有地址检测功能的 9 位模式

此模式通常用于 RS-485 系统中。按如下步骤设置带有地址检测功能的异步接收操作：

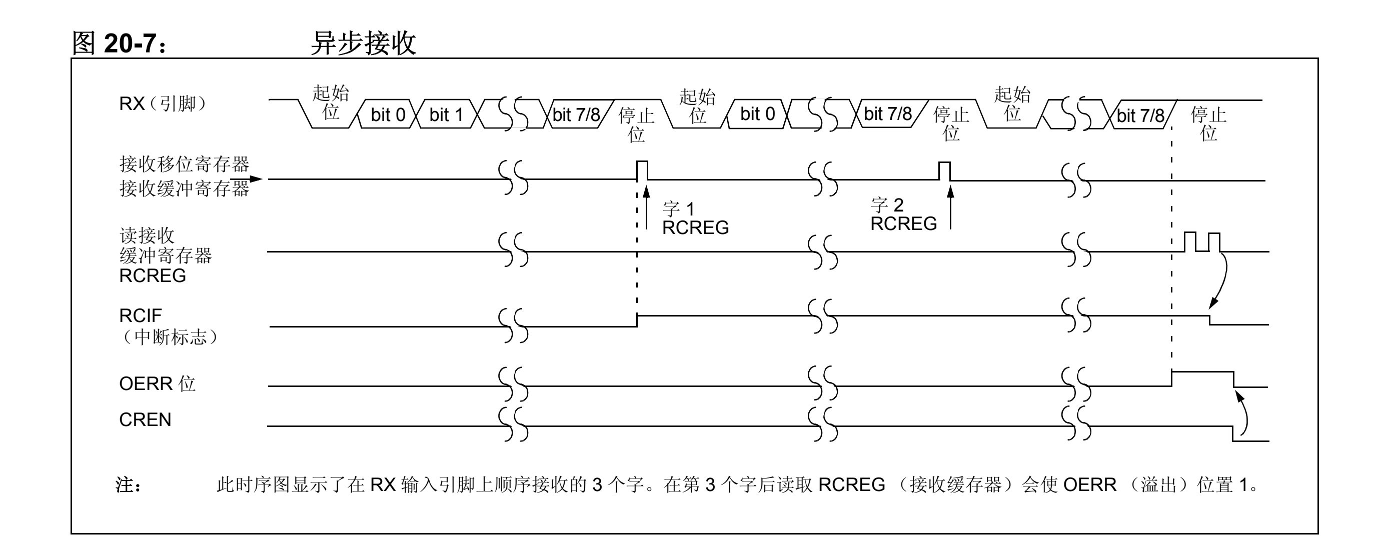
1. 对 SPBRGH:SPBRG 寄存器进行初始化，以设置合适的波特率。按需要将 BRGH 和 BRG16 位置 1 或清零，以获得目标波特率。
2. 通过将 SYNC 位清零并将 SPEN 位置 1 使能异步串口。
3. 若需要中断，请将 RCEN 位置 1 并使用 RCIP 位设置优先级。
4. 将 RX9 位置 1，使能 9 位接收。
5. 将 ADDEN 位置 1，使能地址检测。
6. 将 CREN 位置 1，使能接收。
7. 当接收完成时 RCIF 位将被置 1。此时如果 RCIE 和 GIE 位已置 1，还将响应中断。
8. 读 RCSTA 寄存器判断在接收时是否发生了错误，同时读取第 9 位数据 （如果适用）。
9. 读 RCREG 以判断是否正在对器件进行寻址。
10. 如果发生错误，将 CREN 位清零。
11. 如果已经找到了器件，将 ADDEN 位清零，允许接收到的所有数据进入接收缓冲器，并中断 CPU。

图 20-6： EUSART 接收框图



 初稿

图 20-7： 异步接收



注： 此时序图显示了在 RX 输入引脚上顺序接收的 3 个字。在第 3 个字后读取 RCREG （接收缓存器）会使 OERR （溢出）位置 1。

表 20-6： 与异步接收相关的寄存器

名称	Bit 7	Bit 6	Bit 5	Bit 4	Bit 3	Bit 2	Bit 1	Bit 0	复位值所在页
INTCON	GIE/GIEH	PEIE/GIEL	TMR0IE	INT0IE	RBIE	TMR0IF	INT0IF	RBIF	51
PIR1	SPPIF(1)	ADIF	RCIF	TXIF	SSPIF	CCP1IF	TMR2IF	TMR1IF	54
PIE1	SPPIE(1)	ADIE	RCIE	TXIE	SSPIE	CCP1IE	TMR2IE	TMR1IE	54
IPR1	SPPIP(1)	ADIP	RCIP	TXIP	SSPIP	CCP1IP	TMR2IP	TMR1IP	54
RCSTA	SPEN	RX9	SREN	CREN	ADDEN	FERR	OERR	RX9D	53
RCREG	EUSART 接收寄存器								53
TXSTA	CSRC	TX9	TXEN	SYNC	SENDB	BRGH	TRMT	TX9D	53
BAUDCON	ABDOVF	RCIDL	—	SCKP	BRG16	—	WUE	ABDEN	53
SPBRGH	EUSART 波特率发生器寄存器的高字节								53
SPBRG	EUSART 波特率发生器寄存器的低字节								53

图注： — = 未用 （读为 0）。异步接收不使用阴影单元。

注 1： 在 28 引脚器件上保留；应始终保持这些位清零。

20.2.4 同步间隔字符自动唤醒

在休眠模式下，EUSART 的所有时钟都会暂停。因此，波特率发生器处于非激活状态，且无法进行正确的字节接收。自动唤醒功能允许当RX/DT线上有事件发生时唤醒控制器，该功能需要 EUSART 工作在异步模式下。

通过将 WUE 位 （BAUDCON<1>）置 1，使能自动唤醒功能。该功能启用后，将禁止 RX/DT 上的典型接收操作，且 EUSART 保持在空闲状态并监视唤醒事件 （与 CPU 运行模式无关）。唤醒事件是指 RX/DT 线上发生高电平到低电平的跳变。（这刚好与 “同步间隔” 字符或 LIN 协议唤醒信号字符的启动条件一致。）

唤醒事件后，模块产生一个 RCIF 中断。在正常工作模式下，中断会与 Q 时钟同步产生 （图 20-8）；如果器件处于休眠模式，则两者不同步 （图 20-9） 。通过读 RCREG 寄存器可清除中断条件。

唤醒事件后，当 RX 线上出现由低电平向高电平的跳变时， WUE 位自动清零。此时， EUSART 模块将从空闲状态返回正常工作模式，由此用户可知 “同步间隔” 事件已经结束。

20.2.4.1 使用自动唤醒功能的特别注意事项

因为自动唤醒功能是通过检测RX/DT上的上升沿跳变实现的，所以在停止位前该引脚上的状态改变都可能产生错误的结束信号并导致数据或帧错误。因此，为了确保正确的传输，必须首先发送全 0 字符。对于标准的 RS-232 器件，该字符是 00h （8 位），而对于 LIN 总线器件则是 000h （12 位）。

另外还必须考虑振荡器起振时间，尤其在采用起振延时较长的振荡器 （即 XT 或 HS 模式）的应用中更要注意这一点。“同步间隔”（或唤醒信号）字符必须足够长，并且跟有足够长的时间间隔，以便使振荡器有充足的时间起振并保证 EUSART 正确初始化。

20.2.4.2 使用 WUE 位时的特别注意事项

用 WUE 和 RCIF 事件的时序来判断接收数据的有效性可能会引起混淆。如前所述，将 WUE 位置 1 会使 EUSART 进入空闲模式。唤醒事件会产生一个接收中断，并将 RCIF 位置 1。此后当 RX/DT 出现上升沿时 WUE 位清零。然后通过读 RCREG 寄存器清除中断条件。一般情况下， RCREG 中的数据是无效数据，应该丢弃。

WUE 位清零 （或仍然置 1）且 RCIF 标志位置 1 并不能表明 RCREG 中数据是完整的。用户还应该考虑使用固件验证是否完整地接收了数据。

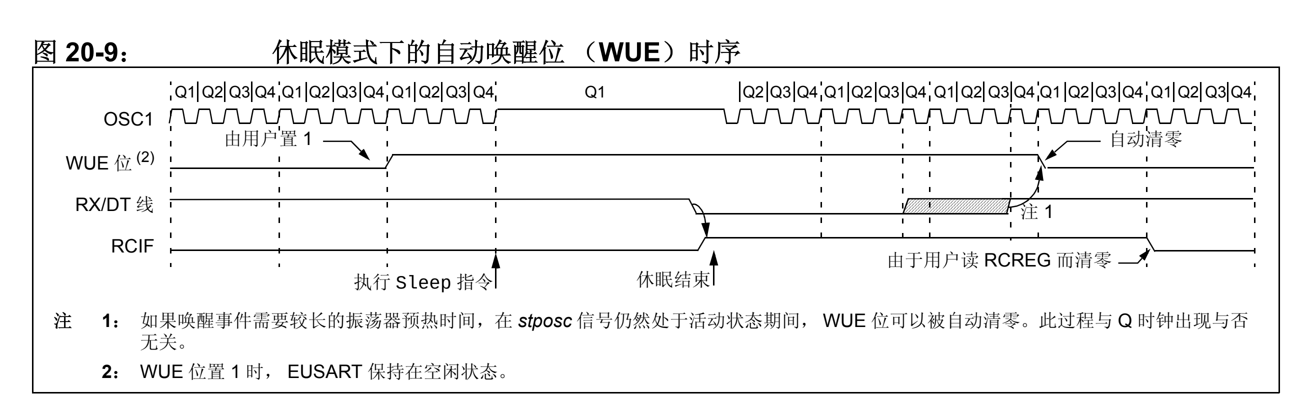
要确保没有丢失有效数据，应检查RCIDL位来验证是否还在接收数据。如果不再进行接收，则可将 WUE 位置 1，使器件立即进入休眠模式。

图 20-8： 正常工作模式下的自动唤醒位 （**WUE**）时序

Q1 Q2 Q3 Q4 Q1 Q2 Q3 Q4 Q1 Q2 Q3 Q4 Q1 Q2 Q3 Q4 Q1 Q2 Q3 Q4 Q1 Q2 Q3 Q4 Q1 Q2 Q3 Q4 Q1 Q2 Q3 Q4 Q1 Q2 Q3 Q4 Q1 Q2 Q3 Q4
OSC1
由用户置 1
自动清零
WUE 位 (1)
RX/DT 线
RCIF
由于用户读 RCREG 而清零

注 1： WUE 位置 1 时， EUSART 保持在空闲状态。

图 20-9： 休眠模式下的自动唤醒位 （**WUE**）时序



注 1： 如果唤醒事件需要较长的振荡器预热时间，在 *stposc* 信号仍然处于活动状态期间， WUE 位可以被自动清零。此过程与 Q 时钟出现与否无关。

2： WUE 位置 1 时， EUSART 保持在空闲状态。

 初稿

20.2.5 间隔字符序列

增强型 EUSART 模块能够发送符合 LIN 总线标准的特殊间隔字符序列。发送的间隔字符包括 1 个起始位，后面跟有 12 个 0 位和一个停止位。当发送移位寄存器装有数据时，只要 SENDB 和 TXEN 位（TXSTA<3> 和 TXSTA<5>）置 1，就会发送帧间隔字符。请注意写入 TXREG 的数据值会被忽略，并会发送全 0。

在发送了停止位后，硬件会自动将 SENDB 位清零。这样用户可以在发送完间隔字符（在 LIN 规范中通常是同步字符）后将下一个要发送的字节预先装入发送 FIFO。

请注意发送间隔字符时写入 TXREG 的数据值会被忽略。写入仅仅是为了启动正确的序列。

如其在正常发送过程中一样，TRMT 位表明发送正在进行还是处于空闲状态。关于间隔字符时序，请参见图 20-10。

20.2.5.1 间隔和同步发送序列

在下面的步骤中会发送一个报文帧头，包括一个间隔字符和其后的自动波特率同步字节。此序列适用于典型的 LIN 总线主器件。

1. 将 EUSART 配置为所需的模式。
2. 将 TXEN 和 SENDB 位置 1，以设置间隔字符。
3. 将无效字符装入 TXREG，启动发送（该值会被忽略）。
4. 将 55h 写入 TXREG，以便把同步字符装入发送 FIFO 缓冲器。
5. 间隔字符发送后，硬件会将 SENDB 位复位。此时，同步字符会以预先配置的模式发送。

当 TXIF 指出 TXREG 为空以后，下一个数据字节会被写入 TXREG。

20.2.6 接收间隔字符

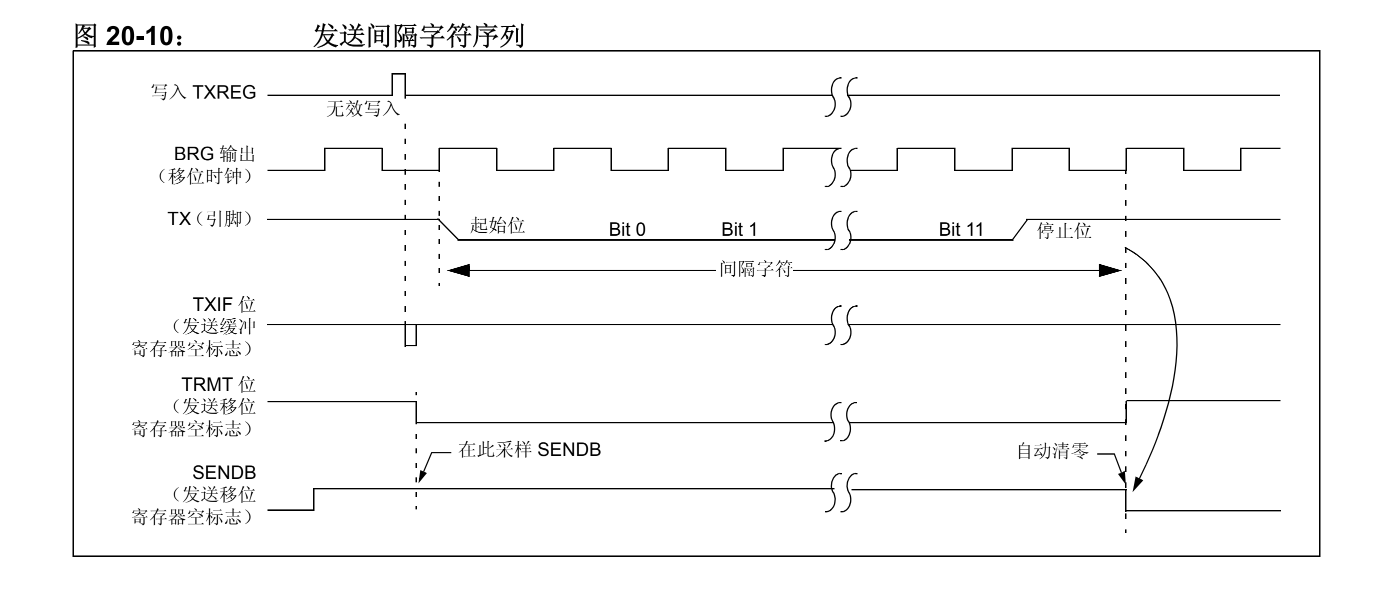
增强型 USART 模块接收间隔字符有两种方法。

第一种方法是强制将波特率配置为典型速率的 9/13。这可以使停止位在正确的采样点（对于间隔字符为起始位之后的 13 位，对于典型数据则是 8 个数据位）产生。

第二种方法使用**第 20.2.4 节“同步间隔字符自动唤醒”**中描述的自动唤醒功能。通过使能此功能，EUSART 将采样 RX/DT 引脚上电平的下两次跳变，产生一个 RCIF 中断，接收下一个数据字节，并在随后产生另一个中断。

请注意在间隔字符后，用户通常希望使能自动波特率检测功能。无论使用哪种方法，用户都可以在检测到 TXIF 中断时马上将 ABD 位置 1。

图 20-10： 发送间隔字符序列



20.3 EUSART 同步主控模式

将 CSRC 位 （TXSTA<7>）置 1 可以进入同步主控模式。在此模式中，数据以半双工方式（即发送和接收不同时进行）发送。发送数据时，禁止接收，反之亦然。将 SYNC 位 （TXSTA<4>）置 1 可进入同步模式。此外，应将使能位 SPEN （RCSTA<7>）置 1，把 TX 和 RX 引脚分别配置为 CK （时钟）和 DT （数据）线。

主控模式意味着处理器在 CK 时钟线上发送主控时钟信号。时钟极性是通过 SCKP 位（BAUDCON<4>）选择的。将 SCKP 置 1 是将空闲状态时的 CK 设为高电平，将该位清零则将空闲状态时的 CK 设为低电平。此选项支持将本模块与 Microwire 器件配合使用。

20.3.1 EUSART 同步主控发送

图 20-3 显示了 EUSART 发送器的框图。发送器的核心是发送（串行）移位寄存器 （TSR）。移位寄存器从读 / 写发送缓冲寄存器 TXREG 中获取数据，而 TXREG 寄存器中的数据由软件装入。在前一次装入数据的最后一位发送完成后，才会向 TSR 寄存器装入新数据。一旦最后一位发送完成，就会将 TXREG 寄存器中的新数据（如果有的话）装入 TSR。

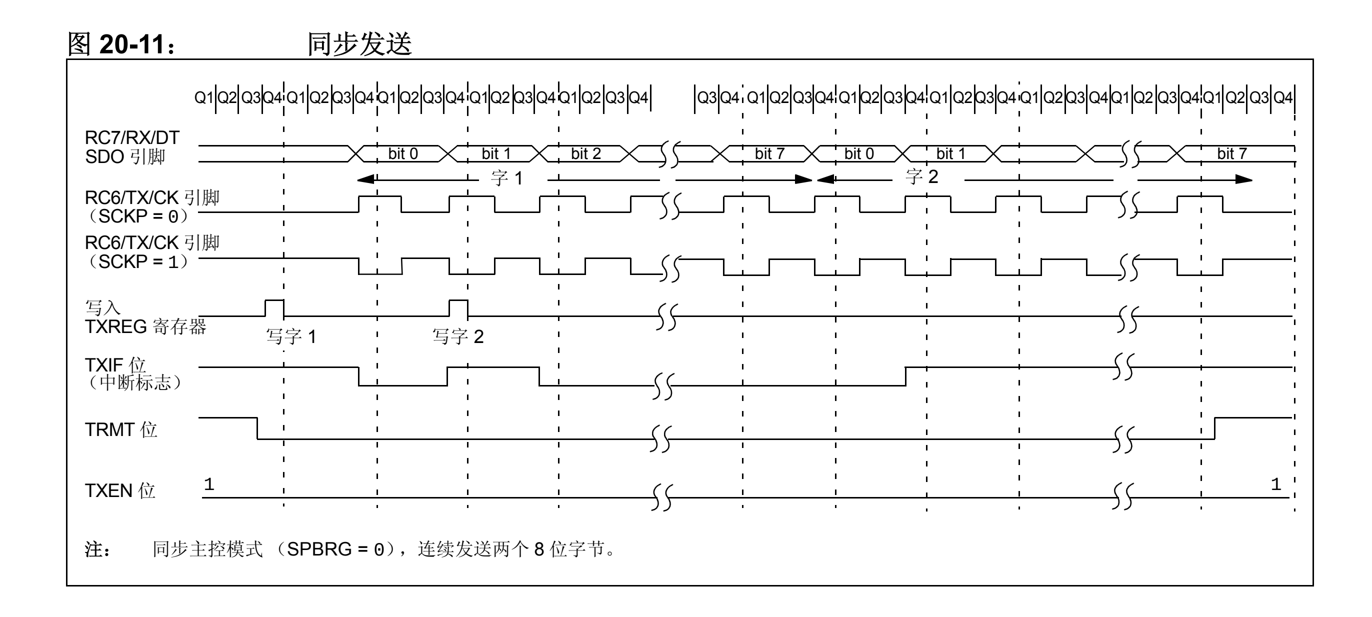
一旦 TXREG 寄存器向 TSR 寄存器传输了数据 （在 1 个 TCY 内发生）， TXREG 寄存器就为空，同时标志位 TXIF（PIR1<4>）被置 1。可以通过将中断允许位 TXIE （PIE1<4>）置 1 或清零来允许 / 禁止该中断。不管 TXIE 位的状态如何，只要中断发生，TXIF 就会置 1 并且不能用软件清零。只有在新数据写入 TXREG 寄存器时，TXIF 才会复位。

TXIF 标志位表示的是 TXREG 寄存器的状态，而另一个标志位 TRMT （TXSTA<1>）则表示 TSR 寄存器的状态。 TRMT 是只读位，它在 TSR 寄存器为空时被置 1。TRMT 位与任何中断均无关联，因此要判断 TSR 寄存器是否为空，用户只能对此位进行轮询。TSR 寄存器并未映射到数据存储器中，因此用户不能直接访问它。

设置同步主控发送操作的步骤如下：

1. 对SPBRGH:SPBRG寄存器进行初始化，以设置合适的波特率。按需要将 BRG16 位置 1 或清零，以获得目标波特率。
2. 通过将 SYNC、SPEN 和 CSRC 位置 1，使能同步主控串口。
3. 如果需要中断，将中断允许位 TXIE 置 1。
4. 若需要发送 9 位数据，将 TX9 位置 1。
5. 将 TXEN 位置 1，使能发送。
6. 如果选择发送9位数据，将第9位数据装入TX9D位。
7. 将数据装入 TXREG 寄存器，启动发送。
8. 若想使用中断，请确保将 INTCON 寄存器中的 GIE 和 PEIE 位 （INTCON<7:6>）置 1。

图 20-11： 同步发送



 初稿

图 20-12： 同步发送 （由 TXEN 位控制）

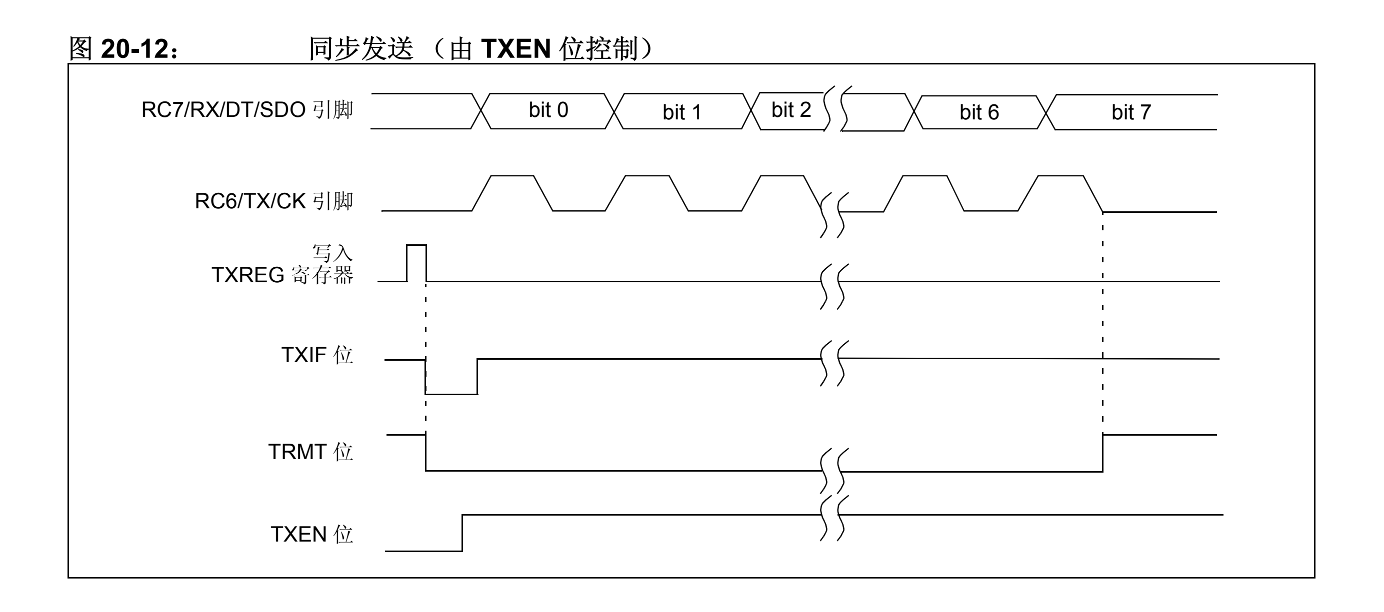


表 20-7： 与同步主控发送相关的寄存器

名称	Bit 7	Bit 6	Bit 5	Bit 4	Bit 3	Bit 2	Bit 1	Bit 0	复位值所在页
INTCON	GIE/GIEH	PEIE/GIEL	TMR0IE	INT0IE	RBIE	TMR0IF	INT0IF	RBIF	51
PIR1	SPPIF(1)	ADIF	RCIF	TXIF	SSPIF	CCP1IF	TMR2IF	TMR1IF	54
PIE1	SPPIE(1)	ADIE	RCIE	TXIE	SSPIE	CCP1IE	TMR2IE	TMR1IE	54
IPR1	SPPIP(1)	ADIP	RCIP	TXIP	SSPIP	CCP1IP	TMR2IP	TMR1IP	54
RCSTA	SPEN	RX9	SREN	CREN	ADDEN	FERR	OERR	RX9D	53
TXREG	EUSART 发送寄存器								53
TXSTA	CSRC	TX9	TXEN	SYNC	SENDB	BRGH	TRMT	TX9D	53
BAUDCON	ABDOVF	RCIDL	—	SCKP	BRG16	—	WUE	ABDEN	53
SPBRGH	EUSART 波特率发生器寄存器的高字节								53
SPBRG	EUSART 波特率发生器寄存器的低字节								53

图注： — = 未用 （读为 0）。同步主控发送不使用阴影单元。

注 1： 在 28 引脚器件上保留；应始终保持这些位清零。

 初稿

20.3.2 EUSART 同步主控接收

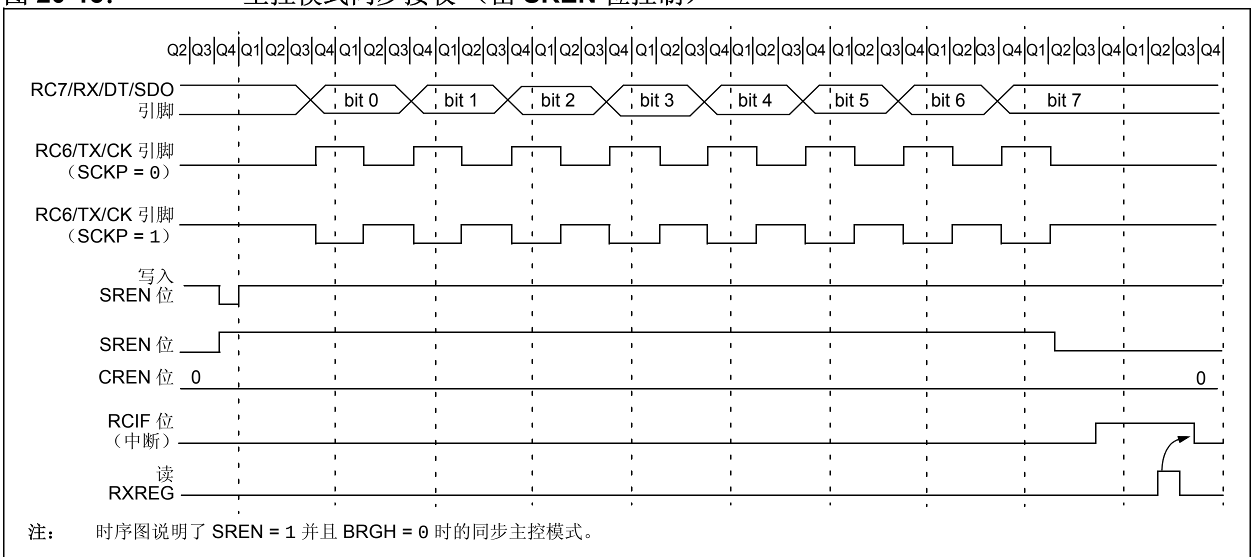
一旦选择了同步模式，只要将单字节接收使能位 SREN（RCSTA<5>）或连续接收使能位 CREN（RCSTA<4>）置 1，即可使能接收。在时钟的下降沿采样 RX 引脚上的数据。

如果将使能位 SREN 置 1，则只接收单个字。如果将使能位 CREN 置 1，则会连续接收数据，直到将 CREN 位清零。如果两个位均被置 1，则 CREN 具有优先权。

设置同步主控接收操作的步骤如下：

1. 对 SPBRGH:SPBRG 寄存器进行初始化，以设置合适的波特率。按需要将 BRG16 位置 1 或清零，以获得目标波特率。
2. 通过将 SYNC、SPEN 和 CSRC 位置 1，使能同步主控串口。
3. 确保将 CREN 和 SREN 位清零。
4. 如果需要中断，将允许位 RCIE 置 1。
5. 若需要接收 9 位数据，将 RX9 位置 1。
6. 若需要单字节接收，将 SREN 位置 1；若需要连续接收，将 CREN 位置 1。
7. 当接收完成时中断标志位 RCIF 将置 1，此时如果允许位 RCIE 已置 1，还将产生一个中断。
8. 读 RCSTA 寄存器获取第 9 位数据（如果已使能），并判断在接收过程中是否发生了错误。
9. 通过读RCREG寄存器来读取接收到的8位数据。
10. 如果发生错误，通过将 CREN 位清零来清除错误。
11. 若想使用中断，请确保将 INTCON 寄存器中的 GIE 和 PEIE 位（INTCON<7:6>）置 1。


图 20-13： 主控模式同步接收（由 SREN 位控制）



注： 时序图说明了 SREN = 1 并且 BRGH = 0 时的同步主控模式。

表 20-8： 与同步主控接收相关的寄存器

名称	Bit 7	Bit 6	Bit 5	Bit 4	Bit 3	Bit 2	Bit 1	Bit 0	复位值所在页
INTCON	GIE/GIEH	PEIE/GIEL	TMR0IE	INT0IE	RBIE	TMR0IF	INT0IF	RBIF	51
PIR1	SPPIF(1)	ADIF	RCIF	TXIF	SSPIF	CCP1IF	TMR2IF	TMR1IF	54
PIE1	SPPIE(1)	ADIE	RCIE	TXIE	SSPIE	CCP1IE	TMR2IE	TMR1IE	54
IPR1	SPPIP(1)	ADIP	RCIP	TXIP	SSPIP	CCP1IP	TMR2IP	TMR1IP	54
RCSTA	SPEN	RX9	SREN	CREN	ADDEN	FERR	OERR	RX9D	53
RCREG	EUSART 接收寄存器								53
TXSTA	CSRC	TX9	TXEN	SYNC	SENDB	BRGH	TRMT	TX9D	53
BAUDCON	ABDOVF	RCIDL	—	SCKP	BRG16	—	WUE	ABDEN	53
SPBRGH	EUSART 波特率发生器寄存器的高字节								53
SPBRG	EUSART 波特率发生器寄存器的低字节								53

图注： — = 未用（读为 0）。同步主控接收不使用阴影单元。
注 1： 在 28 引脚器件上保留；应始终保持这些位清零。

 初稿

20.4 EUSART 同步从动模式

将 CSRC （TXSTA<7>）位清零可进入同步从动模式。此模式与同步主控模式的区别在于移位时钟由 CK 引脚上的外部时钟提供 （主控模式下由内部时钟提供）。这使得器件能在任何低功耗模式下发送或接收数据。

20.4.1 EUSART 同步从动发送

除了休眠模式以外，同步主控、从动模式的工作方式是完全相同的。

如果向 TXREG 寄存器写入两个字，然后执行 SLEEP 指令，则将发生以下事件：

a) 第一个字立即传送到 TSR 寄存器进行发送。
b) 第二个字仍保留在 TXREG 寄存器中。
c) 不会将标志位 TXIF 置 1。
d) 当第一个字移出 TSR 后，TXREG 寄存器将把第二个字送入 TSR，同时将标志位 TXIF 置 1。
e) 如果中断允许位 TXIE 已置 1，中断将把器件从休眠状态唤醒。如果允许了全局中断，程序则会跳转到中断向量处执行。

设置同步从动发送操作的步骤如下：

1. 通过将 SYNC 和 SPEN 位置 1 并将 CSRC 位清零使能同步从动串口。
2. 将 CREN 和 SREN 位清零。
3. 如果需要中断，将中断允许位 TXIE 置 1。
4. 若需要发送 9 位数据，将 TX9 位置 1。
5. 将中断允许位 TXEN 置 1，使能发送。
6. 如果选择发送9位数据，将第9位数据装入TX9D位。
7. 将数据装入 TXREG 寄存器，启动发送。
8. 若想使用中断，请确保将 INTCON 寄存器中的 GIE 和 PEIE 位 （INTCON<7:6>）置 1。

表 20-9： 与同步从动发送相关的寄存器

名称	Bit 7	Bit 6	Bit 5	Bit 4	Bit 3	Bit 2	Bit 1	Bit 0	复位值所在页
INTCON	GIE/GIEH	PEIE/GIEL	TMR0IE	INT0IE	RBIE	TMR0IF	INT0IF	RBIF	51
PIR1	SPPIF[(1)]	ADIF	RCIF	TXIF	SSPIF	CCP1IF	TMR2IF	TMR1IF	54
PIE1	SPPIE[(1)]	ADIE	RCIE	TXIE	SSPIE	CCP1IE	TMR2IE	TMR1IE	54
IPR1	SPPIP[(1)]	ADIP	RCIP	TXIP	SSPIP	CCP1IP	TMR2IP	TMR1IP	54
RCSTA	SPEN	RX9	SREN	CREN	ADDEN	FERR	OERR	RX9D	53
TXREG	EUSART 发送寄存器								53
TXSTA	CSRC	TX9	TXEN	SYNC	SENDB	BRGH	TRMT	TX9D	53
BAUDCON	ABDOVF	RCIDL	—	SCKP	BRG16	—	WUE	ABDEN	53
SPBRGH	EUSART 波特率发生器寄存器的高字节								53
SPBRG	EUSART 波特率发生器寄存器的低字节								53

图注： — = 未用 （读为 0）。同步从动发送不使用阴影单元。

注 1： 在 28 引脚器件上保留；应始终保持这些位清零。

20.4.2 EUSART 同步从动接收

除了休眠模式、空闲模式以及在从动模式下忽略 SREN 位以外，同步主控和同步从动模式的工作方式完全相同。

如果在进入休眠或空闲模式前将 CREN 位置 1，使能接收，那么在该低功耗模式下可以接收到一个数据字。接收到该字后，RSR 寄存器将把数据传输到 RCREG 寄存器，如果中断允许位 RCIE 已置 1，产生的中断将把器件从低功耗模式唤醒。如果允许了全局中断，程序则会跳转到中断向量处执行。

设置同步从动接收操作的步骤如下：

1. 通过将 SYNC 和 SPEN 位置 1 并将 CSRC 位清零使能同步主控串口。
2. 如果需要中断，将允许位 RCIE 置 1。
3. 若需要接收 9 位数据，将 RX9 位置 1。
4. 将使能位 CREN 置 1，使能接收。
5. 当接收完成时，RCIF 标志位将被置 1。如果允许位 RCIE 已置 1，还将产生一个中断。
6. 读 RCSTA 寄存器获取第 9 位数据（如果已使能），并判断在接收过程中是否发生了错误。
7. 通过读RCREG寄存器来读取接收到的8位数据。
8. 如果发生错误，通过将 CREN 位清零来清除错误。
9. 若想使用中断，请确保将 INTCON 寄存器中的 GIE 和 PEIE 位（INTCON<7:6>）置 1。

表 20-10： 与同步从动接收相关的寄存器

名称	Bit 7	Bit 6	Bit 5	Bit 4	Bit 3	Bit 2	Bit 1	Bit 0	复位值所在页
INTCON	GIE/GIEH	PEIE/GIEL	TMR0IE	INT0IE	RBIE	TMR0IF	INT0IF	RBIF	51
PIR1	SPPIF[(1)]	ADIF	RCIF	TXIF	SSPIF	CCP1IF	TMR2IF	TMR1IF	54
PIE1	SPPIE[(1)]	ADIE	RCIE	TXIE	SSPIE	CCP1IE	TMR2IE	TMR1IE	54
IPR1	SPPIP[(1)]	ADIP	RCIP	TXIP	SSPIP	CCP1IP	TMR2IP	TMR1IP	54
RCSTA	SPEN	RX9	SREN	CREN	ADDEN	FERR	OERR	RX9D	53
RCREG	EUSART 接收寄存器								53
TXSTA	CSRC	TX9	TXEN	SYNC	SENDB	BRGH	TRMT	TX9D	53
BAUDCON	ABDOVF	RCIDL	—	SCKP	BRG16	—	WUE	ABDEN	53
SPBRGH	EUSART 波特率发生器寄存器的高字节								53
SPBRG	EUSART 波特率发生器寄存器的低字节								53

图注： — = 未用（读为 0）。同步从动接收不使用阴影单元。

注 1： 在 28 引脚器件上保留；应始终保持这些位清零。

 初稿

21.0 10 位模数转换器 （A/D）模块

28 引脚器件的模数 （Analog-to-Digital， A/D）转换器具有 10 路输入，而 40/44 引脚器件的模数转换器则具有 13 路输入。 A/D 模块能将一个模拟输入信号转换成相应的 10 位数字信号。

此模块有 5 个寄存器：

- A/D 转换结果高位寄存器 （ADRESH）
- A/D 转换结果低位寄存器 （ADRESL）
- A/D 转换控制寄存器 0 （ADCON0）
- A/D 转换控制寄存器 1 （ADCON1）
- A/D 转换控制寄存器 2 （ADCON2）

如寄存器 21-1 所示，A/D 模块的工作方式由 ADCON0 寄存器控制。如寄存器 21-2 所示，端口引脚的功能由 ADCON1 寄存器配置。如寄存器 21-3 所示，由 ADCON2 寄存器对 A/D 时钟源、编程采集时间和输出结果的对齐方式进行配置。

寄存器 21-1： ADCON0：A/D 控制寄存器 0

U-0	U-0	R/W-0	R/W-0	R/W-0	R/W-0	R/W-0	R/W-0
—	—	CHS3	CHS2	CHS1	CHS0	GO/$\overline{\text{DONE}}$	ADON
bit 7							bit 0

bit 7-6 未用：读为 0

bit 5-2 **CHS3:CHS0**：模拟通道选择位

0000 = 通道 0 （AN0）
0001 = 通道 1 （AN1）
0010 = 通道 2 （AN2）
0011 = 通道 3 （AN3）
0100 = 通道 4 （AN4）
0101 = 通道 5 （AN5）[1,2]
0110 = 通道 6 （AN6）[1,2]
0111 = 通道 7 （AN7）[1,2]
1000 = 通道 8 （AN8）
1001 = 通道 9 （AN9）
1010 = 通道 10 （AN10）
1011 = 通道 11 （AN11）
1100 = 通道 12 （AN12）
1101 = 未用通道 [2]
1110 = 未用通道 [2]
1111 = 未用通道 [2]

注 1：这些通道在 28 引脚器件上未用。
2：在未用通道上执行转换会返回引脚悬空输入的测量值。

bit 1 **GO/$\overline{\text{DONE}}$**：A/D 转换状态位

当 ADON = 1 时：
1 = A/D 转换正在进行
0 = A/D 空闲

bit 0 **ADON**：A/D 模块使能位

1 = 使能 A/D 转换模块
0 = 禁止 A/D 转换模块

图注：			
R = 可读位	W = 可写位	U = 未用位，读为 0	
-n = 上电复位时的值	1 = 置 1	0 = 清零	x = 未知

寄存器 21-2： ADCON1：A/D 控制寄存器 1

U-0	U-0	R/W-0	R/W-0	R/W-0[(1)]	R/W[(1)]	R/W[(1)]	R/W[(1)]
—	—	VCFG1	VCFG0	PCFG3	PCFG2	PCFG1	PCFG0
bit 7							bit 0

bit 7-6 未用：读为 0

bit 5 **VCFG1**：参考电压配置位 （VREF- 电压源）
1 = VREF- （AN2）
0 = VSS

bit 4 **VCFG0**：参考电压配置位 （VREF+ 电压源）
1 = VREF+ （AN3）
0 = VDD

bit 3-0 **PCFG3:PCFG0**：A/D 端口配置控制位：

PCFG3: PCFG0	AN12	AN11	AN10	AN9	AN8	AN7[(2)]	AN6[(2)]	AN5[(2)]	AN4	AN3	AN2	AN1	AN0
0000[(1)]	A	A	A	A	A	A	A	A	A	A	A	A	A
0001	A	A	A	A	A	A	A	A	A	A	A	A	A
0010	A	A	A	A	A	A	A	A	A	A	A	A	A
0011	D	A	A	A	A	A	A	A	A	A	A	A	A
0100	D	D	A	A	A	A	A	A	A	A	A	A	A
0101	D	D	D	A	A	A	A	A	A	A	A	A	A
0110	D	D	D	D	A	A	A	A	A	A	A	A	A
0111[(1)]	D	D	D	D	D	A	A	A	A	A	A	A	A
1000	D	D	D	D	D	D	A	A	A	A	A	A	A
1001	D	D	D	D	D	D	D	A	A	A	A	A	A
1010	D	D	D	D	D	D	D	D	A	A	A	A	A
1011	D	D	D	D	D	D	D	D	D	A	A	A	A
1100	D	D	D	D	D	D	D	D	D	D	A	A	A
1101	D	D	D	D	D	D	D	D	D	D	D	A	A
1110	D	D	D	D	D	D	D	D	D	D	D	D	A
1111	D	D	D	D	D	D	D	D	D	D	D	D	D

A = 模拟输入 D = 数字 I/O

注 **1**： PCFG 位的上电复位值取决于 PBADEN 配置位的值。当 PBADEN = 1 时，PCFG<3:0> = 0000；当 PBADEN = 0 时， PCFG<3:0> = 0111。

2： 仅 40/44 引脚器件具有 AN5 到 AN7 通道。

图注：			
R = 可读位	W = 可写位	U = 未用位，读为 0	
-n = 上电复位时的值	1 = 置 1	0 = 清零	x = 未知

寄存器 21-3： ADCON2：A/D 控制寄存器 2

R/W-0	U-0	R/W-0	R/W-0	R/W-0	R/W-0	R/W-0	R/W-0
ADFM	—	ACQT2	ACQT1	ACQT0	ADCS2	ADCS1	ADCS0
bit 7							bit 0

bit 7 **ADFM**：A/D 转换结果格式选择位
1 = 右对齐
0 = 左对齐

bit 6 未用：读为 0

bit 5-3 **ACQT2:ACQT0**：A/D 采集时间选择位
111 = 20 TAD
110 = 16 TAD
101 = 12 TAD
100 = 8 TAD
011 = 6 TAD
010 = 4 TAD
001 = 2 TAD
000 = 0 TAD(1)

bit 2-0 **ADCS2:ADCS0**：A/D 转换时钟选择位
111 = FRC （时钟来自 A/D 模块的 RC 振荡器）(1)
110 = FOSC/64
101 = FOSC/16
100 = FOSC/4
011 = FRC （时钟来自 A/D 模块的 RC 振荡器）(1)
010 = FOSC/32
001 = FOSC/8
000 = FOSC/2

注 1： 如果选择了 FRC 时钟源，在 A/D 时钟启动之前会加上一个 TCY （指令周期）的延时。这可以保证在开始转换之前执行 SLEEP 指令。

图注：			
R = 可读位	W = 可写位	U = 未用位，读为 0	
-n = 上电复位时的值	1 = 置 1	0 = 清零	x = 未知

可用软件选择模拟参考电压为器件的正电源电压和负电源电压（VDD 和 VSS）或 RA3/AN3/VREF+ 引脚和 RA2/AN2/VREF-/CVREF 引脚上的电平。

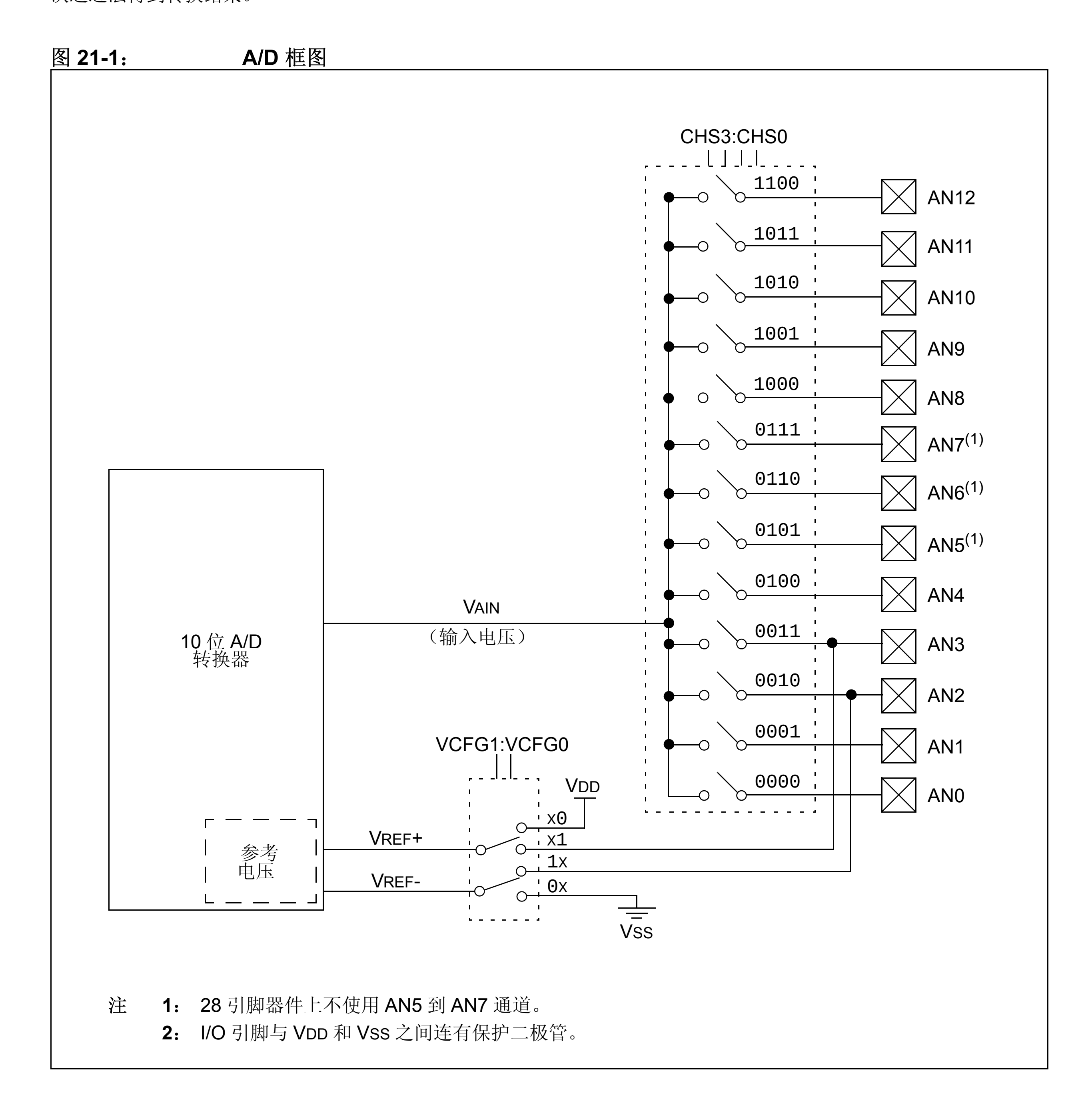
A/D 转换器具备可在休眠模式下工作的特性。要使 A/D 转换器在休眠状态下工作， A/D 转换时钟必须来自于 A/D 模块内部的 RC 振荡器。

采样保持电路的输出作为转换器的输入，转换器采用逐次逼近法得到转换结果。

器件复位强制所有寄存器进入复位状态，同时强制关闭 A/D 模块并中止任何正在进行的转换。

可以将每个与 A/D 转换器相关的端口引脚配置为模拟输入或数字 I/O 引脚。 ADRESH 和 ADRESL 寄存器保存 A/D 转换的结果。当 A/D 转换完成之后，转换结果被装入 ADRESH:ADRESL 寄存器对， GO/DONE 位（ADCON0 寄存器）清零，且 A/D 中断标志位 ADIF 被置 1。 A/D 模块的框图如图 21-1。

图 21-1： A/D 框图



注 1： 28 引脚器件上不使用 AN5 到 AN7 通道。
2： I/O 引脚与 VDD 和 VSS 之间连有保护二极管。

 初稿

上电复位时， ADRESH:ADRESL 寄存器中的值保持不变。上电复位后，ADRESH:ADRESL 寄存器中的值不确定。

在根据需要配置好 A/D 模块之后，必须在转换开始之前对选定的通道进行采样。必须将模拟输入通道相应的 TRIS 位选择为输入。采集时间的确定请参见**第 21.1 节 "A/D 采集要求"**。采集时间一结束，即可启动 A/D 转换。可将采集时间编程设定在 GO/DONE 位置 1 和实际转换启动之间。

执行 A/D 转换时应该遵循以下步骤：

1. 配置 A/D 模块：
   - 配置模拟引脚、参考电压和数字 I/O （通过 ADCON1 寄存器）
   - 选择 A/D 输入通道 （通过 ADCON0 寄存器）
   - 选择 A/D 采集时间 （通过 ADCON2 寄存器）
   - 选择 A/D 转换时钟 （通过 ADCON2 寄存器）
   - 使能 A/D 模块 （通过 ADCON0 寄存器）
2. 需要时，配置 A/D 中断：
   - 清零 ADIF 位
   - 将 ADIE 位置 1
   - 将 GIE 位置 1
3. 如果需要，等待所需的采集时间。
4. 启动转换：
   - 将 GO/DONE 位置 1（通过 ADCON0 寄存器）
5. 等待 A/D 转换完成，通过以下两种方法之一判断转换是否完成：
   - 查询 GO/DONE 位是否被清零
   - 等待 A/D 中断
6. 读取 A/D 结果寄存器（ADRESH:ADRESL），需要时将 ADIF 位清零。
7. 如需再次进行 A/D 转换，返回步骤 1 或步骤 2。将每位的 A/D 转换时间定义为 T_{AD}，在下一次采集开始前至少需要等待 3 个 T_{AD}。

图 21-2： A/D 传递函数

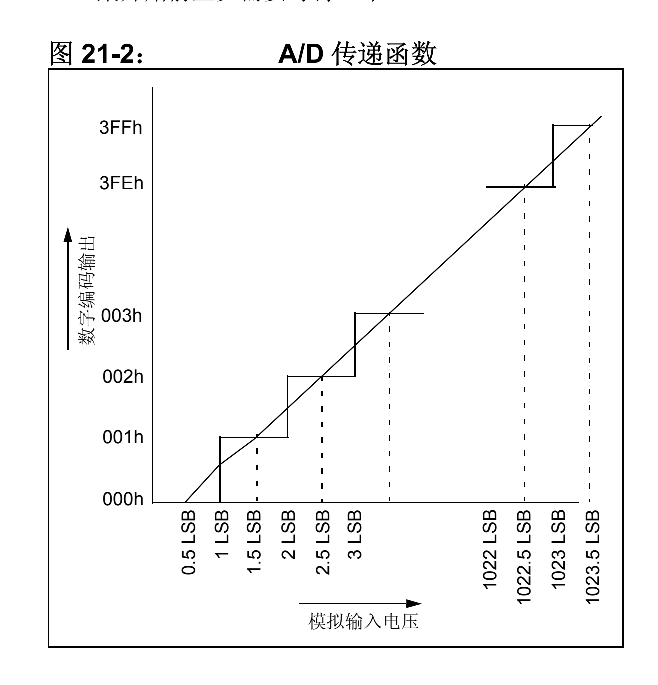
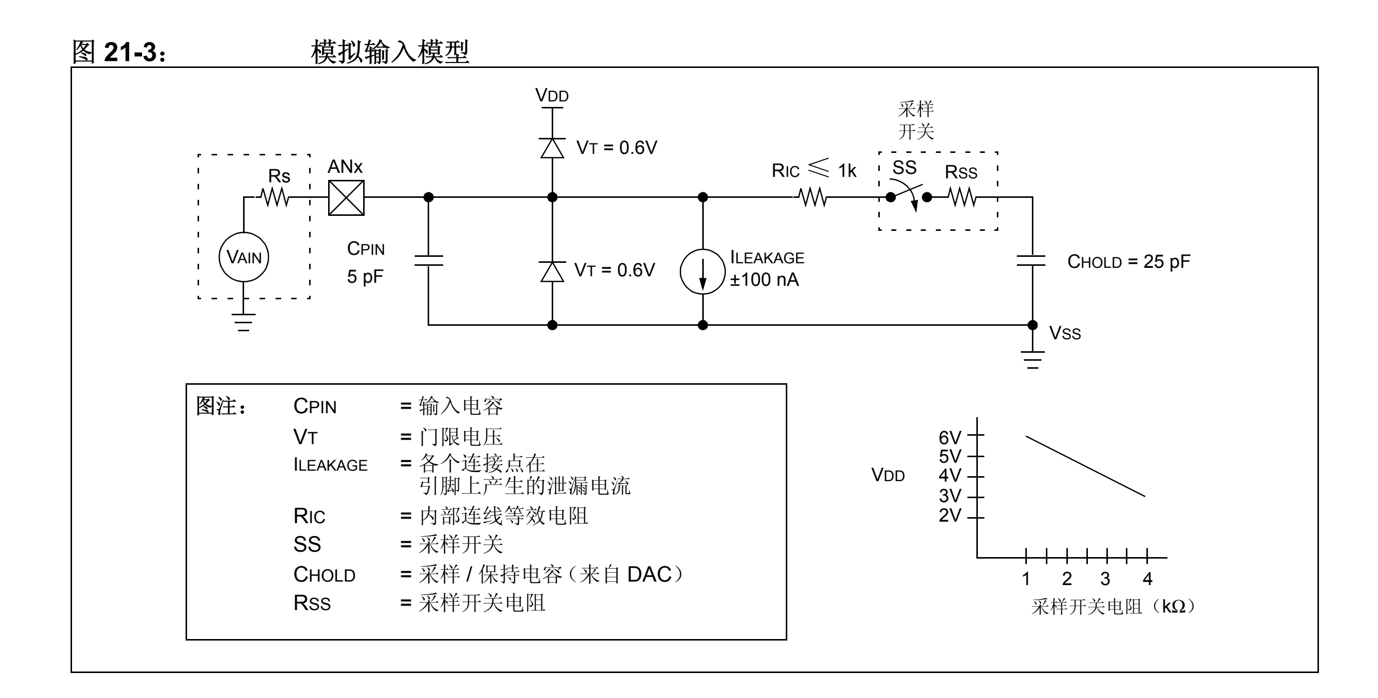


图 21-3： 模拟输入模型



21.1 A/D 采集要求

为了使 A/D 转换器达到规定精度，必须使充电保持电容（CHOLD）充满至输入通道的电平。图 21-3 给出了模拟输入的电路模型。电源阻抗 （RS）和内部采样开关阻抗（RSS）直接影响为电容 CHOLD 充电所需的时间。采样开关阻抗值（RSS）随器件电压（VDD）不同而改变。电源阻抗将影响模拟输入的失调电压（由于引脚泄漏电流的原因）。**模拟信号源的最大阻抗推荐值为 2.5 kΩ**。在选择（改变）了模拟输入通道之后，必须对通道进行采样才能启动转换，采样时间必须大于最小采集时间。

注：	当开始转换时，要将保持电容与输入引脚断开。

可使用公式 21-1 来计算最小采集时间。该公式假定的误差为 1/2 LSb（A/D 转换需要 1024 步）。1/2 LSb 的误差是 A/D 模块达到规定分辨率所能允许的最大误差。

例 21-3 说明了所需的最小采集时间 TACQ 的计算过程。计算结果基于以下假设：

CHOLD	=	25 pF
Rs	=	2.5 kΩ
转换误差	≤	1/2 LSb
VDD	=	5V → Rss = 2 kΩ
温度	=	85°C （系统最大值）

公式 21-1： 采集时间

$$T_{ACQ} = \text{放大器稳定时间} + \text{保持电容充电时间} + \text{温度系数}$$
$$= T_{AMP} + T_C + T_{COFF}$$

公式 21-2： A/D 最小充电时间

$$V_{HOLD} = (V_{REF} – (V_{REF}/2048)) \cdot (1 – e^{(-T_C/C_{HOLD}(R_{IC} + R_{SS} + R_S))})$$

或

$$T_C = -(C_{HOLD})(R_{IC} + R_{SS} + R_S) \ln(1/2048)$$

公式 21-3： 计算所需的最小采集时间

$$T_{ACQ} = T_{AMP} + T_C + T_{COFF}$$

$$T_{AMP} = 0.2\mu s$$

$$T_{COFF} = (Temp – 25°C)(0.02\ \mu s/°C)$$
$$= (50° C – 25°C)(0.02\ \mu s/°C)$$
$$= 1.2\ \mu s$$

只有在温度 > 25°C 时需要温度系数。当温度低于 25°C 时， T_{COFF} = 0 ms。

$$T_C = -(C_{HOLD})(R_{IC} + R_{SS} + R_S) \ln(1/2047)\ \mu s$$
$$= -(25\ pF) (1\ k\Omega + 2\ k\Omega + 2.5\ k\Omega) \ln(0.0004883)\ \mu s$$
$$= 5.03\ \mu s$$

$$T_{ACQ} = 0.2\ \mu s + 5\ \mu s + 1.2\ \mu s$$
$$= 6.4\ \mu s$$

 初稿

21.2 选择和配置采集时间

ADCON2 寄存器允许用户选择采集时间，该时间在每当 GO/DONE 位置 1 时发生。该寄存器还提供了自动设定采集时间的选项。

可以使用 ACQT2:ACQT0 位（ADCON2<5:3>）设置采集时间，采集时间的范围是2到20 个TAD。当GO/DONE 位置 1 时，A/D 模块继续对输入进行采样，采样时间为所选择的采集时间，然后自动启动转换。由于采集时间已被编程，因此在选择通道和 GO/DONE 位置 1 之间无需另外等待一个采集时间。

若 ACQT2:ACQT0 = 000，则表示选择手动采集。当 GO/DONE 位置 1 时，采样停止并启动转换。用户应确保在选定所需要的输入通道之后到 GO/DONE 位置 1 之间经过了所需要的采集时间。此选项也是 ACQT2:ACQT0 位的默认复位状态，并且与不提供可编程采集时间的器件兼容。

在这两种情况下，当转换完成时，GO/DONE位被清零、ADIF 标志位被置 1 且 A/D 再次开始对当前选定的通道进行采样。如果采集时间已经被编程，那么将不会有任何标志显示采集时间何时结束，转换何时开始。

21.3 选择 A/D 转换时钟

每位的 A/D 转换时间定义为 TAD。每完成一次 10 位 A/D 转换需要 11 个 TAD 。可用软件选择 A/D 转换时钟源。TAD 有以下 7 种可能的选择：

- 2 TOSC
- 4 TOSC
- 8 TOSC
- 16 TOSC
- 32 TOSC
- 64 TOSC
- 内部 RC 振荡器

为了实现正确的 A/D 转换， A/D 转换时钟（TAD）必须尽可能小，但它必须大于最小 TAD（如需了解更多信息，请参见表 28-29 中的参数 130）。

表 21-1给出了器件在不同工作频率下和选择不同的A/D 时钟源时得到的 TAD。

表 21-1： 不同器件工作频率下的 TAD

AD 时钟源 （TAD）		最高器件频率	
工作频率	ADCS2:ADCS0	PIC18FXXXX	PIC18LFXXXX[4]
2 TOSC	000	2.86 MHz	1.43 kHz
4 TOSC	100	5.71 MHz	2.86 MHz
8 TOSC	001	11.43 MHz	5.72 MHz
16 TOSC	101	22.86 MHz	11.43 MHz
32 TOSC	010	40.0 MHz	22.86 MHz
64 TOSC	110	40.0 MHz	22.86 MHz
RC[3]	x11	1.00 MHz[1]	1.00 MHz[2]

注 1： RC 时钟源的典型 TAD 时间为 4ms。
2： RC 时钟源的典型 TAD 时间为 6ms。
3： 当器件工作频率高于 1 MHz 时，整个转换过程必须在休眠模式下进行，否则 A/D 精度可能超出规范允许的范围。
4： 仅适用于低功耗器件。

21.4 在功耗管理模式下的工作方式

处于功耗管理模式时，自动采集时间以及 A/D 转换时钟的选择一定程度上可由时钟源和频率决定。

如果希望在处于功耗管理模式时进行 A/D 转换，ADCON2 中的 ACQT2:ACQT0 和 ADCS2:ADCS0 位就应该根据该模式下使用的时钟源进行更新。在进入功耗管理模式之后，就可以开始 A/D 采集或转换。一旦开始采集或转换，器件应继续使用相同的时钟源直到转换完成。

如果需要，器件也可以在转换过程中被置于相应的空闲模式。如果器件时钟频率小于 1 MHz，就应该选择 A/D 模块的 RC 时钟源。

在休眠模式下工作需要选择 A/D 模块的 FRC 时钟。如果 ACQT2:ACQT0 位被置为 000 并启动转换，转换将被延迟一个指令周期以允许执行 SLEEP 指令并进入休眠模式。IDLEN 位（OSCCON<7>）必须在转换开始之前被清零。

21.5 配置模拟端口引脚

ADCON1、TRISA、TRISB 和 TRISE 寄存器均可用于配置 A/D 端口引脚。必须将配置为模拟输入的端口引脚对应的 TRIS 位置 1（输入）。如果将 TRIS 位清零（输出），则该引脚将输出数字电平（VOH 或 VOL）。

A/D 转换操作与 CHS3:CHS0 位以及 TRIS 位的状态无关。

> 注 **1：** 读取端口寄存器时，所有配置为模拟输入通道的引脚均读为 0（低电平）。配置为数字输入的引脚将模拟输入电平精确转换为数字电平。
>
> **2：** 定义为数字输入引脚上的模拟电平，可能会导致数字输入缓冲器消耗的电流超出器件规范。
>
> **3：** 配置寄存器 3H 中的 PBADEN 位通过控制 ADCON1 中的 PCFG0 位的复位状态，可以将 PORTB 引脚配置为在复位时作为模拟引脚或数字引脚。

21.6 A/D 转换

图 21-4 显示了在 GO/$\overline{DONE}$ 位置 1 且 ACQT2:ACQT0 位被清零后 A/D 转换器的工作状态。转换在下一条指令执行之后开始，以允许器件在转换开始之前进入休眠模式。

图 21-5 显示了在 GO/$\overline{DONE}$ 位置 1 且 ACQT2:CQT0 位被设置为 010（即在转换开始之前选择了 4 TAD 的采集时间）后 A/D 转换器的工作状态。

在转换期间将 GO/$\overline{DONE}$ 位清零将中止当前的 A/D 转换。不会用尚未完成的 A/D 转换结果更新 A/D 结果寄存器对。这意味着 ADRESH:ADRESL 寄存器将仍然保存上一次的转换结果（或上一次写入 ADRESH:ADRESL 寄存器的值）。

在 A/D 转换完成或中止以后，需要等待 2 个 TAD 才能开始下一次采集。等待时间一到，将自动开始对选定通道进行采集。

注：	不能用启动 A/D 模块的同一指令将 GO/$\overline{DONE}$ 位置 1。

21.7 放电

放电过程用于对电容阵列的值进行初始化。在每次采样之前都会对此阵列放电。因为电路总是需要根据当前采样的值对该电容阵列充电，而不是根据以前测量的值进行充放电，所以这一特性有助于优化单位增益放大器。

图 21-4： A/D 转换 TAD 周期 （ACQT<2:0> = 000， TACQ = 0）

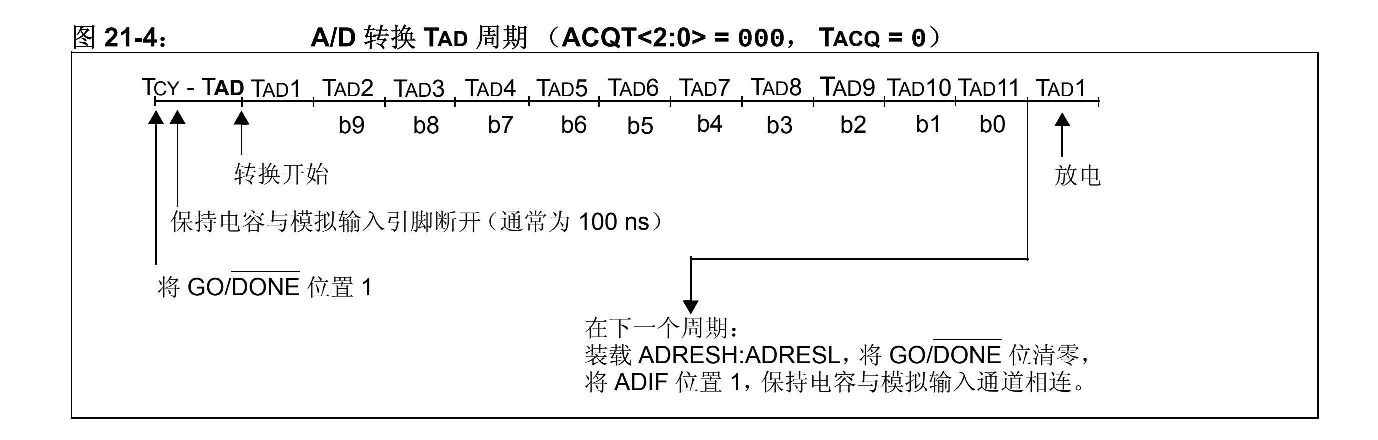
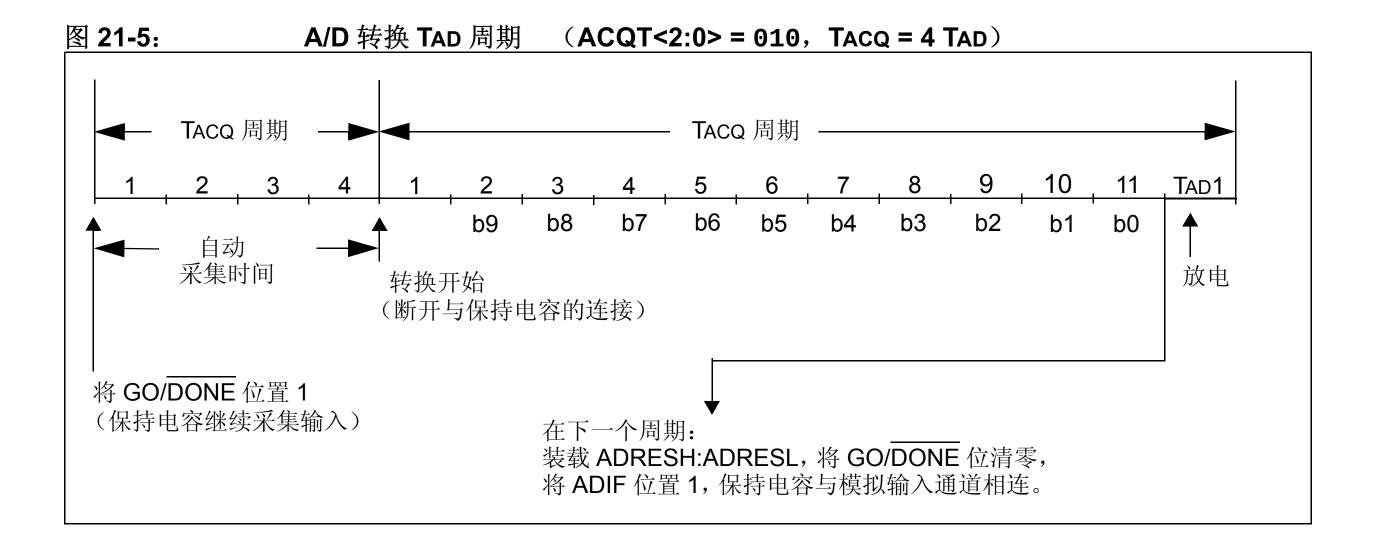


图 21-5： A/D 转换 TAD 周期 （ACQT<2:0> = 010，TACQ = 4 TAD）



21.8 CCP2 触发信号的使用

CCP2 模块的特殊事件触发信号可以启动 A/D 转换。这需要将 CCP2M3:CCP2M0 位（CCP2CON<3:0>）设置为 1011，且使能 A/D 模块（ADON 位置 1）。发生触发事件时，GO/DONE 位将被置 1，启动 A/D 采样和转换并将 Timer1（或 Timer3）计数器复位为 0。复位 Timer1（或 Timer3）可自动重复 A/D 采集，最大限度地降低了软件开销（将 ADRESH:ADRESL 内容移到目标单元）。在特殊事件触发信号将 GO/DONE 位置 1（启动转换）前，用户必须选择正确的模拟输入通道，并通过手动或选择相应的 TACQ 时间来设定最小采集时间。

如果未使能 A/D 模块（ADON 清零），则特殊事件触发信号对 A/D 模块不产生任何影响，但它仍会将 Timer1（或 Timer3）计数器复位。

表 21-2：与 A/D 操作相关的寄存器

名称	Bit 7	Bit 6	Bit 5	Bit 4	Bit 3	Bit 2	Bit 1	Bit 0	复位值所在页
INTCON	GIE/GIEH	PEIE/GIEL	TMR0IE	INT0IE	RBIE	TMR0IF	INT0IF	RBIF	51
PIR1	SPPIF(4)	ADIF	RCIF	TXIF	SSPIF	CCP1IF	TMR2IF	TMR1IF	54
PIE1	SPPIE(4)	ADIE	RCIE	TXIE	SSPIE	CCP1IE	TMR2IE	TMR1IE	54
IPR1	SPPIP(4)	ADIP	RCIP	TXIP	SSPIP	CCP1IP	TMR2IP	TMR1IP	54
PIR2	OSCFIF	CMIF	USBIF	EEIF	BCLIF	HLVDIF	TMR3IF	CCP2IF	54
PIE2	OSCFIE	CMIE	USBIE	EEIE	BCLIE	HLVDIE	TMR3IE	CCP2IE	54
IPR2	OSCFIP	CMIP	USBIP	EEIP	BCLIP	HLVDIP	TMR3IP	CCP2IP	54
ADRESH	A/D 结果寄存器的高字节								52
ADRESL	A/D 结果寄存器的低字节								52
ADCON0	—	—	CHS3	CHS2	CHS1	CHS0	GO/DONE	ADON	52
ADCON1	—	—	VCFG1	VCFG0	PCFG3	PCFG2	PCFG1	PCFG0	52
ADCON2	ADFM	—	ACQT2	ACQT1	ACQT0	ADCS2	ADCS1	ADCS0	52
PORTA	—	RA6(2)	RA5	RA4	RA3	RA2	RA1	RA0	54
TRISA	—	TRISA6(2)	TRISA5	TRISA4	TRISA3	TRISA2	TRISA1	TRISA0	54
PORTB	RB7	RB6	RB5	RB4	RB3	RB2	RB1	RB0	54
TRISB	TRISB7	TRISB6	TRISB5	TRISB4	TRISB3	TRISB2	TRISB1	TRISB0	54
LATB	LATB7	LATB6	LATB5	LATB4	LATB3	LATB2	LATB1	LATB0	54
PORTE	RDPU(4)	—	—	—	RE3(1,3)	RE2(4)	RE1(4)	RE0(4)	54
TRISE(4)	—	—	—	—	—	TRISE2	TRISE1	TRISE0	54
LATE(4)	—	—	—	—	—	LATE2	LATE1	LATE0	54

图注： — = 未用（读为 0）。A/D 转换不使用阴影单元。

注 1： 仅当主复位功能被禁止（MCLRE 配置位 = 0）时使用。

2： 根据振荡器配置将 RA6 及其相关的锁存器和数据方向位设置为 I/O 引脚，否则，它们将读为 0。

3： 当 MCLRE 配置位为 0 时，RE3 端口位只能作为输入。

4： 这些寄存器和 / 或位在 28 引脚器件上不存在。

22.0 比较器模块

模拟比较器模块包含两个比较器，可以用多种方式对它们进行配置。比较器的输入可以是与 RA0 到 RA5 引脚复用的模拟输入引脚，也可以为片上参考电压（见**第 23.0 节“比较器参考电压模块”**）。数字输出（正常或翻转的）可从引脚获取也可通过控制寄存器读取。

CMCON 寄存器（寄存器 22-1）用于配置比较器的输入和输出。图 22-1 显示了各种比较器的配置。

寄存器 22-1： CMCON：比较器控制寄存器

R-0	R-0	R/W-0	R/W-0	R/W-0	R/W-1	R/W-1	R/W-1
C2OUT	C1OUT	C2INV	C1INV	CIS	CM2	CM1	CM0
bit 7							bit 0

bit 7 **C2OUT**：比较器 2 输出位
当 C2INV = 0 时：
1 = C2 VIN+ > C2 VIN-
0 = C2 VIN+ < C2 VIN-
当 C2INV = 1 时：
1 = C2 VIN+ < C2 VIN-
0 = C2 VIN+ > C2 VIN-

bit 6 **C1OUT**：比较器 1 输出位
当 C1INV = 0 时：
1 = C1 VIN+ > C1 VIN-
0 = C1 VIN+ < C1 VIN-
当 C1INV = 1 时：
1 = C1 VIN+ < C1 VIN-
0 = C1 VIN+ > C1 VIN-

bit 5 **C2INV**：比较器 2 输出翻转位
1 = C2 输出翻转
0 = C2 输出不翻转

bit 4 **C1INV**：比较器 1 输出翻转位
1 = C1 输出翻转
0 = C1 输出不翻转

bit 3 **CIS**：比较器输入开关位
当 CM2:CM0 = 110 时：
1 = C1 VIN- 连接到 RA3/AN3
C2 VIN- 连接到 RA2/AN2
0 = C1 VIN- 连接到 RA0/AN0
C2 VIN- 连接到 RA1/AN1

bit 2-0 **CM2:CM0**：比较器模式位
图 22-1 给出了比较器的几种模式以及相应 CM2:CM0 位的设置。

图注：			
R = 可读位	W = 可写位	U = 未用位，读为 0	
-n = 上电复位时的值	1 = 置 1	0 = 清零	x = 未知

22.1 比较器配置

图 22-1 给出了比较器的 8 种工作模式。CMCON 寄存器的 CM2:CM0 位用于选择这些模式。TRISA 寄存器控制每种模式下比较器引脚的数据方向。如果改变比较器模式，由于存在特定的模式改变延迟（如第 **28.0** 节"电气规范"所示），比较器可能会在此延迟期间输出无效电平。

注：	改变比较器工作模式期间应该禁止比较器中断，以免产生错误中断。

图 22-1： 比较器 I/O 工作模式

比较器复位
CM2:CM0 = 000
RA0/AN0 A VIN- ; RA3/AN3/VREF+ A VIN+ ; C1 关（读为 0）
RA1/AN1 A VIN- ; RA2/AN2/VREF-/CVREF A VIN+ ; C2 关（读为 0）

比较器关闭（上电复位时的默认值）
CM2:CM0 = 111
RA0/AN0 D VIN- ; RA3/AN3/VREF+ D VIN+ ; C1 关（读为 0）
RA1/AN1 D VIN- ; RA2/AN2/VREF-/CVREF D VIN+ ; C2 关（读为 0）

两个独立的比较器
CM2:CM0 = 010
RA0/AN0 A VIN- ; RA3/AN3/VREF+ A VIN+ ; C1 C1OUT
RA1/AN1 A VIN- ; RA2/AN2/VREF-/CVREF A VIN+ ; C2 C2OUT

两个独立的带输出的比较器
CM2:CM0 = 011
RA0/AN0 A VIN- ; RA3/AN3/VREF+ A VIN+ ; C1 C1OUT
RA4/T0CKI/C1OUT*/RCV
RA1/AN1 A VIN- ; RA2/AN2/VREF-/CVREF A VIN+ ; C2 C2OUT
RA5/AN4/SS/HLVDIN/C2OUT*

两个具有公共参考端的比较器
CM2:CM0 = 100
RA0/AN0 A VIN- ; RA3/AN3/VREF+ A VIN+ ; C1 C1OUT
RA1/AN1 A VIN- ; RA2/AN2/VREF-/CVREF D VIN+ ; C2 C2OUT

两个具有公共参考端且带输出的比较器
CM2:CM0 = 101
RA0/AN0 A VIN- ; RA3/AN3/VREF+ A VIN+ ; C1 C1OUT
RA4/T0CKI/C1OUT*/RCV
RA1/AN1 A VIN- ; RA2/AN2/VREF-/CVREF D VIN+ ; C2 C2OUT
RA5/AN4/SS/HLVDIN/C2OUT*

一个带输出的独立比较器
CM2:CM0 = 001
RA0/AN0 A VIN- ; RA3/AN3/VREF+ A VIN+ ; C1 C1OUT
RA4/T0CKI/C1OUT*/RCV
RA1/AN1 D VIN- ; RA2/AN2/VREF-/CVREF D VIN+ ; C2 关（读为 0）

两个比较器复用四路输入
CM2:CM0 = 110
RA0/AN0 A ; RA3/AN3/VREF+ A ; CIS = 0 ; CIS = 1 ; VIN- ; VIN+ ; C1 C1OUT
RA1/AN1 A ; RA2/AN2/VREF-/CVREF A ; CIS = 0 ; CIS = 1 ; VIN- ; VIN+ ; C2 C2OUT
CVREF 来自 VREF 模块

A = 模拟输入，始终读为 0　D = 数字输入　CIS（CMCON<3>）是比较器输入开关
* 将 TRISA<5:4> 位置 1 会通过把引脚配置为输入引脚而禁止比较器输出。

　初稿　

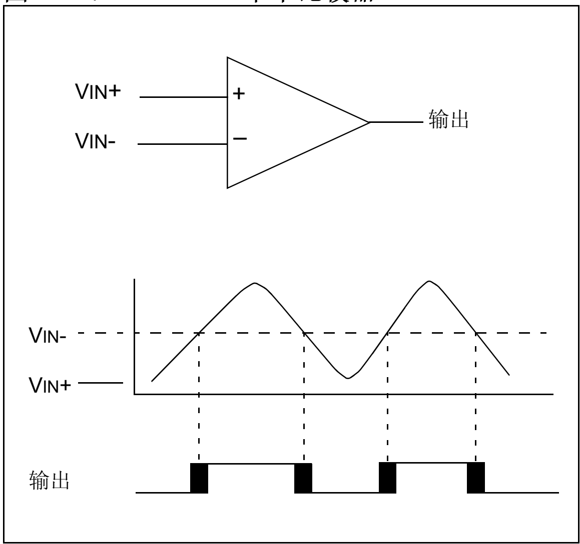
22.2 比较器工作原理

图 22-2 所示为单个比较器及其模拟输入电平和数字输出之间的关系。如果 VIN+ 上的模拟输入电平小于 VIN- 上的模拟输入电平，那么比较器将输出数字低电平。当 VIN+ 上的模拟输入电平高于 VIN- 上的模拟输入电平时，比较器输出数字高电平。图 22-2 中比较器输出的阴影部分表示因输入失调和响应时间所造成的输出不确定区。

22.3 比较器参考电压

根据比较器的工作模式，可选择使用外部或内部参考电压。将 VIN- 上的模拟信号和 VIN+ 上的模拟信号作比较，并相应的调整比较器的数字输出 （图 22-2）。

图 22-2： 单个比较器



22.3.1 外部参考电压信号

当使用外部参考电压时，可以将比较器模块中的两个比较器配置为使用同一个参考源或使用不同的参考源。然而，门限检测电路可能要求使用同一个参考源。参考信号必须介于 VSS 和 VDD 之间，且可被施加到比较器的任一引脚上。

22.3.2 内部参考电压信号

比较器模块也可以选择使用内部参考电压模块产生的参考电压。在**第 23.0 节 “比较器参考电压模块”**中详细介绍了该模块。

只有在两个比较器复用四路输入的模式（CM2:CM0 = `110`）中才可使用内部参考电压。该模式下，内部参考电压被施加到两个比较器的 VIN+ 引脚上。

22.4 比较器的响应时间

响应时间是指从选定一个新的参考电压或输入源到比较器输出达到一个有效电平的最短时间。如果内部参考电压发生了改变，在使用比较器的输出时必须考虑到内部参考电压的最大延时。否则，应该使用比较器的最大延时 （见**第 28.0 节 “电气规范”**）。

22.5 比较器输出

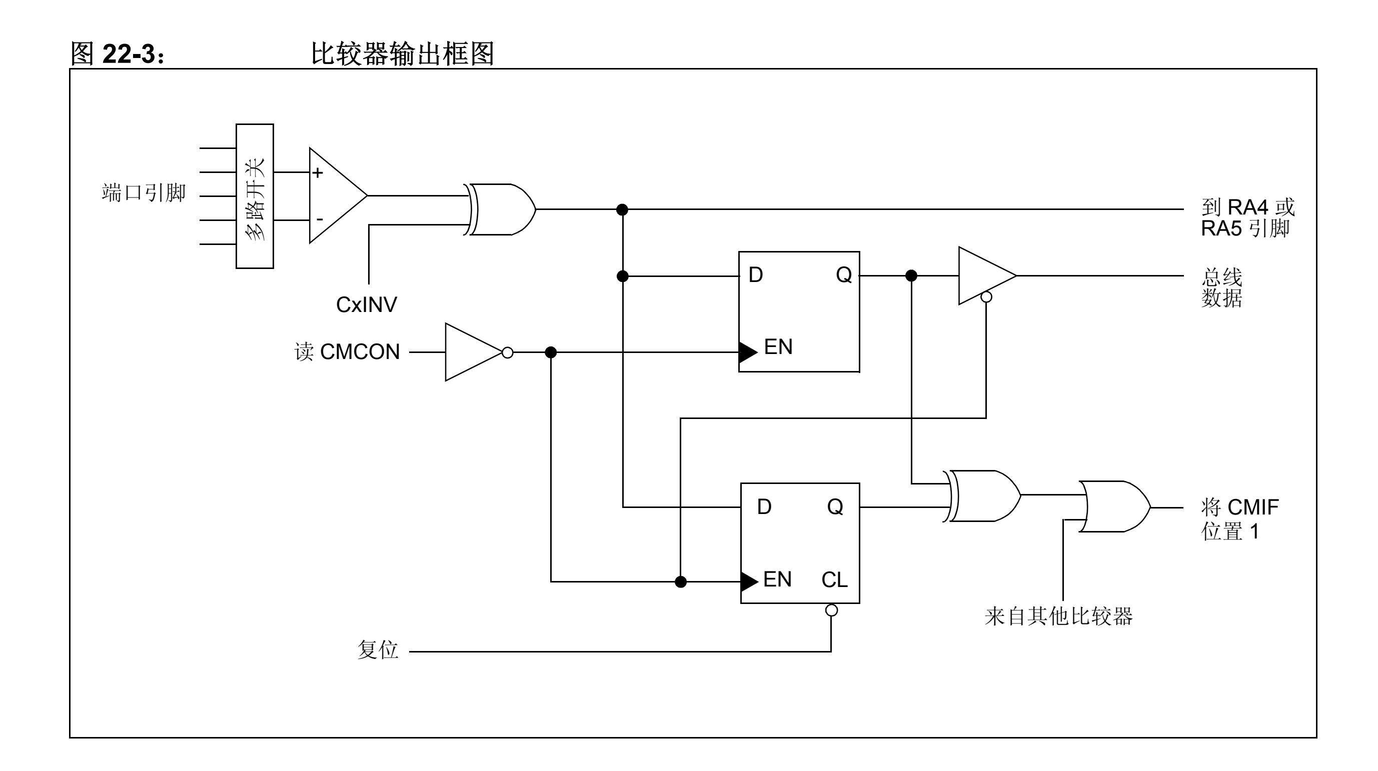
通过 CMCON 寄存器可读取比较器的输出。该寄存器是只读的。比较器的输出也可以直接输出到 I/O 引脚 RA4 和 RA5。当使能时， RA4 和 RA5 引脚输出路径上的多路开关会发生切换，每个引脚输出的信号与比较器输出信号是异步的。每个比较器输出的不确定区域的大小与电气规范中给出的输入失调电压和响应时间有关。图 22-3 为比较器的输出原理框图。

在该模式下，TRISA 位仍作为 RA4 和 RA5 引脚的输出使能 / 禁止位。

设置 C2INV 和 C1INV 位（CMCON<5:4>）可以改变比较器输出电平的极性。

注	
1：	读端口寄存器时，所有配置为模拟输入的引脚将读为 `0`。配置为数字输入的引脚根据施密特触发器输入规范，对模拟输入信号进行相应的转换。
2：	施加在数字输入引脚上的模拟电平会使其输入缓冲器消耗的电流超过规定值。

图 22-3： 比较器输出框图



22.6 比较器中断

任何一个比较器的输出电平一旦发生了变化，就会将该比较器的中断标志位置 1。为确定比较器输出是否发生了变化，需要用软件来保持输出位的状态信息（从 CMCON<7:6> 中读取数据）。CMIF 位（PIR2<6>）位是比较器中断标志位，且必须通过清零复位。因为可以人为向 CMCON 寄存器写入 1，所以也可以实现模拟中断。

必须将 CMIE 位（PIE2<6>）和 PEIE 位（INTCON<6>）置1以允许中断。此外，也必须置位GIE（INTCON<7>）位。只要这些位中的任何一位被清零，虽然当有中断条件产生时 CMIF 位仍会置 1，但中断是被禁止的。

注：	当执行读操作时（在 Q2 周期开始时），如果 CMCON 寄存器（C1OUT 或 C2OUT）的值发生了变化，那么 CMIF（PIR 寄存器）中断标志位可能不会被置 1。

用户可用以下方式在中断服务程序中清除该中断：

a) 对CMCON的任何读或写操作均将结束电平不匹配状态。

b) 将中断标志位 CMIF 清零。

引脚上电平的不匹配会不断地将 CMIF 标志位置 1。读 CMCON 寄存器将清除中断条件，并允许将 CMIF 标志位清零。

22.7 在休眠模式下比较器的操作

当比较器处于运行模式而器件处于休眠模式时，比较器一直工作并可使用比较器中断（如果使能的话）。当允许中断时，中断会把器件从休眠模式唤醒。每个比较器工作时都会消耗额外的电流，如比较器的规范中所示。若要把休眠模式下的功耗减少到最小，可在进入休眠状态前关闭比较器模块（CM2:CM0 = 111）。当器件从休眠模式被唤醒时， CMCON 寄存器的内容不受影响。

22.8 复位的影响

器件复位强制 CMCON 寄存器进入复位状态，导致比较器模块被关闭（CM2:CM0 = 111）。但是，在器件复位时输入引脚（RA0 到 RA3）被默认配置为模拟输入。PCFG3:PCFG0 位（ADCON1<3:0>）用于对这些引脚进行 I/O 配置。因此，复位时引脚呈现模拟输入状态，器件电流降至最小。

22.9 模拟输入连接注意事项

图 22-4 是一个简化的模拟输入电路。由于模拟引脚和数字输出端相连，因此在模拟引脚与 VDD 和 VSS 之间连有反向偏置的二极管，将其电压限制在 VSS 和 VDD 之间。一旦输入电压超出该范围 0.6V 以上，其中一个二极管就会发生正向偏置从而使输入电压被钳位。模拟信号源的最大阻抗推荐值为 10 kΩ。任何连接到模拟输入引脚上的外部元件 （如电容器或齐纳二极管），要保证其泄漏电流极小。

图 22-4： 比较器模拟输入模型

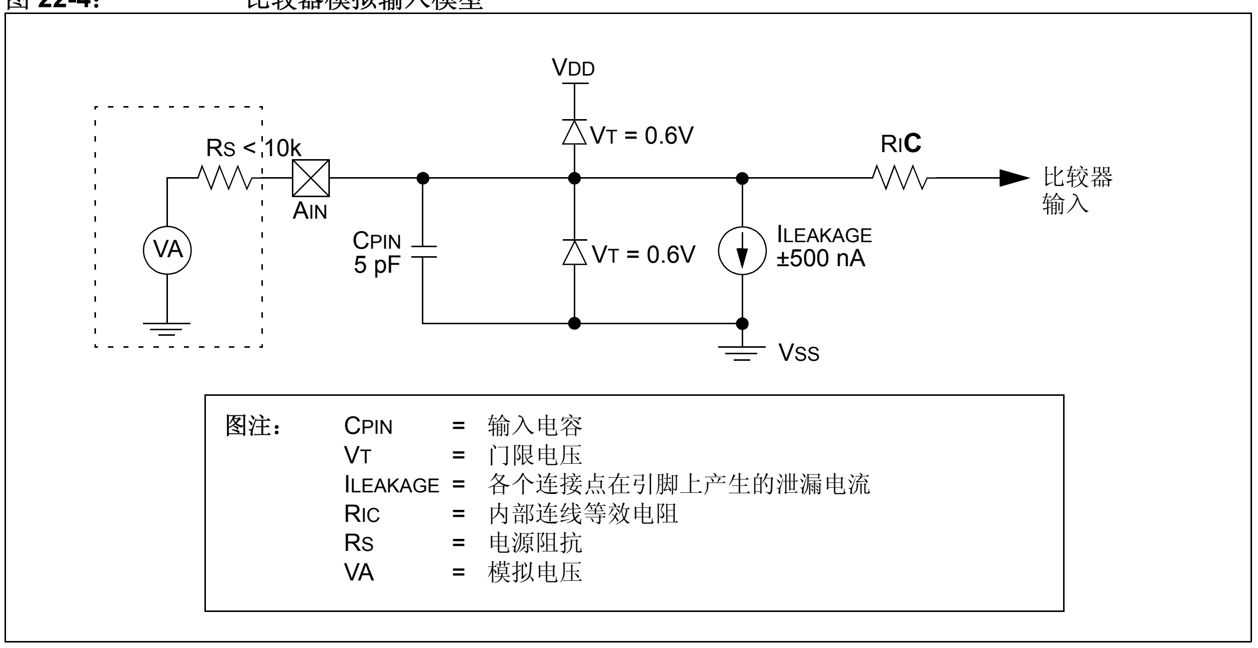


表 22-1： 与比较器模块相关的寄存器

名称	Bit 7	Bit 6	Bit 5	Bit 4	Bit 3	Bit 2	Bit 1	Bit 0	复位值所在页
CMCON	C2OUT	C1OUT	C2INV	C1INV	CIS	CM2	CM1	CM0	53
CVRCON	CVREN	CVROE	CVRR	CVRSS	CVR3	CVR2	CVR1	CVR0	53
INTCON	GIE/GIEH	PEIE/GIEL	TMR0IE	INT0IE	RBIE	TMR0IF	INT0IF	RBIF	51
PIR2	OSCFIF	CMIF	USBIF	EEIF	BCLIF	HLVDIF	TMR3IF	CCP2IF	54
PIE2	OSCFIE	CMIE	USBIE	EEIE	BCLIE	HLVDIE	TMR3IE	CCP2IE	54
IPR2	OSCFIP	CMIP	USBIP	EEIP	BCLIP	HLVDIP	TMR3IP	CCP2IP	54
PORTA	—	RA6[1]	RA5	RA4	RA3	RA2	RA1	RA0	54
LATA	—	LATA6[1]	LATA5	LATA4	LATA3	LATA2	LATA1	LATA0	54
TRISA	—	TRISA6[1]	TRISA5	TRISA4	TRISA3	TRISA2	TRISA1	TRISA0	54

图注： — = 未用 （读为 0）。比较器模块不使用阴影单元。

注 1： 根据各种振荡器模式， PORTA<6> 及其方向和锁存位被单独配置为端口引脚。这些位在禁止时读为 0。

注：

23.0 比较器参考电压模块

比较器参考电压模块是一个 16 阶的梯形电阻网络，提供可选择的参考电压。虽然它的主要作用是给模拟比较器提供参考电压，但也可将它用于其他场合。

图 23-1 为模块的框图。梯形电阻经过分段可提供两种范围的 CVREF 值，并且还具有断电功能以在不使用参考电压时降低功耗。模块的供电参考电压由器件 VDD/VSS 或外部参考电压提供。

23.1 配置比较器参考电压

参考电压模块由 CVRCON 寄存器（寄存器 23-1）控制。比较器参考电压提供两种范围的输出电压，每种范围都具有 16 种不同的电平。CVRR 位（CVRCON<5>）用于选择电压范围。这两种电压范围的主要区别在于 CVREF 选择位（CVR3:CVR0）选定的步长不同，其中一个范围具有更高的分辨率。下面是计算比较器参考电压输出值的公式：

如果 CVRR = 1：
CVREF = ((CVR3:CVR0)/24) x CVRSRC

如果 CVRR = 0：
CVREF = (CVRSRC/4) + (((CVR3:CVR0)/32) x CVRSRC)

比较器参考电压模块供电可以来自 VDD 和 VSS，或者与 RA2 和 RA3 复用的外部 VREF+ 和 VREF-。CVRSS 位（CVRCON<4>）用于选择电压源。

在改变 CVREF 输出值时 必须考虑到比较器参考电压的稳定时间（见第 **28.0节“电气规范”**中的表 28-3）。

寄存器 23-1： CVRCON：比较器参考电压控制寄存器

R/W-0	R/W-0	R/W-0	R/W-0	R/W-0	R/W-0	R/W-0	R/W-0
CVREN	CVROE(1)	CVRR	CVRSS	CVR3	CVR2	CVR1	CVR0
bit 7							bit 0

bit 7 **CVREN**：比较器参考电压使能位
1 = 开启CVREF 电路
0 = 关闭 CVREF 电路

bit 6 **CVROE**：比较器 VREF 输出使能位 (1)
1 = 同样在 RA2/AN2/VREF-/CVREF 引脚上输出 CVREF 电压
0 = CVREF 电压与 RA2/AN2/VREF-/CVREF 引脚断开

注 1：CVROE 的优先级高于 TRISA<2> 位。

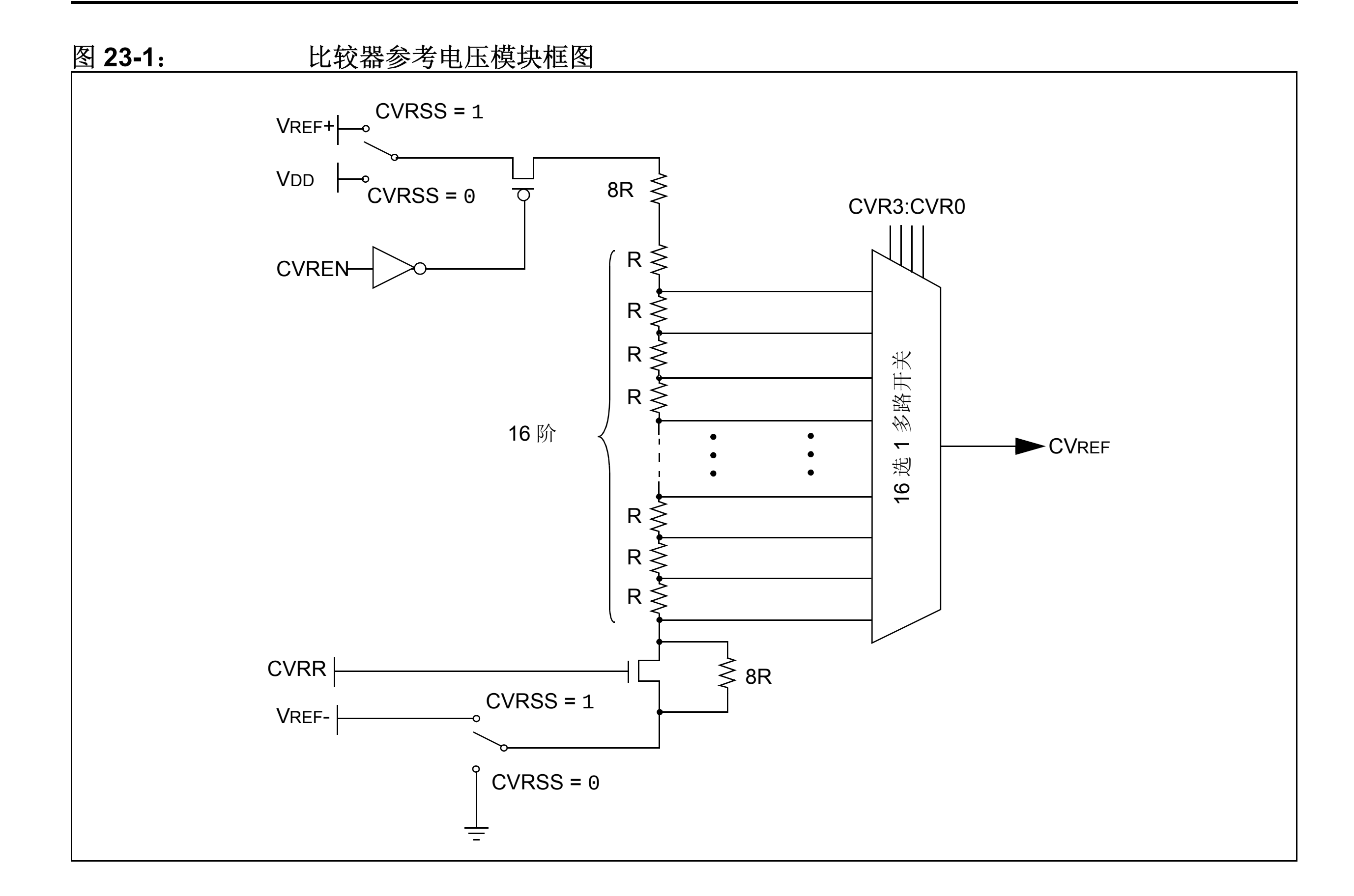
bit 5 **CVRR**：比较器 VREF 范围选择位
1 = 0 到 0.667 CVRSRC，步长为 CVRSRC/24 （低压范围）
0 = 0.25 CVRSRC 到 0.75 CVRSRC，步长为 CVRSRC/32 （高压范围）

bit 4 **CVRSS**：比较器 VREF 源选择位
1 = 比较器参考电压源 CVRSRC = (VREF+)–(VREF-)
0 = 比较器参考电压源 CVRSRC = VDD–VSS

bit 3-0 **CVR3:CVR0**：比较器 VREF 值选择位 （0 ≤ （CVR3:CVR0）≤ 15）
当 CVRR = 1：
CVREF = ((CVR3:CVR0)/24) x CVRSRC
当 CVRR = 0：
CVREF = (CVRSRC/4) + (((CVR3:CVR0)/32) x CVRSRC)

图注：			
R = 可读位	W = 可写位	U= 未用位，读为 0	
-n = 上电复位时的值	1 = 置 1	0 = 清零	x = 未知

图 23-1： 比较器参考电压模块框图



23.2 参考电压精度 / 误差

由于模块结构上的原因，模块并不能实现参考电压源满幅输出。梯形电阻网络顶部和底部的晶体管（图 23-1）使CVREF 无法达到参考电压源的上下限。参考电压来自于参考电压源；因此CVREF输出电平会随参考电压源一起波动。经测试的参考电压绝对精度请参见**第 28.0 节“电气规范”**。

23.3 休眠期间的操作

当中断或看门狗定时器超时唤醒器件时， CVRCON 寄存器内容不受影响。为了最大限度地降低休眠模式下的电流消耗，应关闭参考电压模块。

23.4 复位的影响

器件复位时，CVREN（CVRCON<7>）位将被清零从而禁止参考电压模块。复位还将 CVROE （CVRCON<6>）位清零，从而使参考电压与 RA2 引脚断开；同时通过将 CVRR （CVRCON<5>）位清零选择高压范围。复位也会将 CVR 值选择位清零。

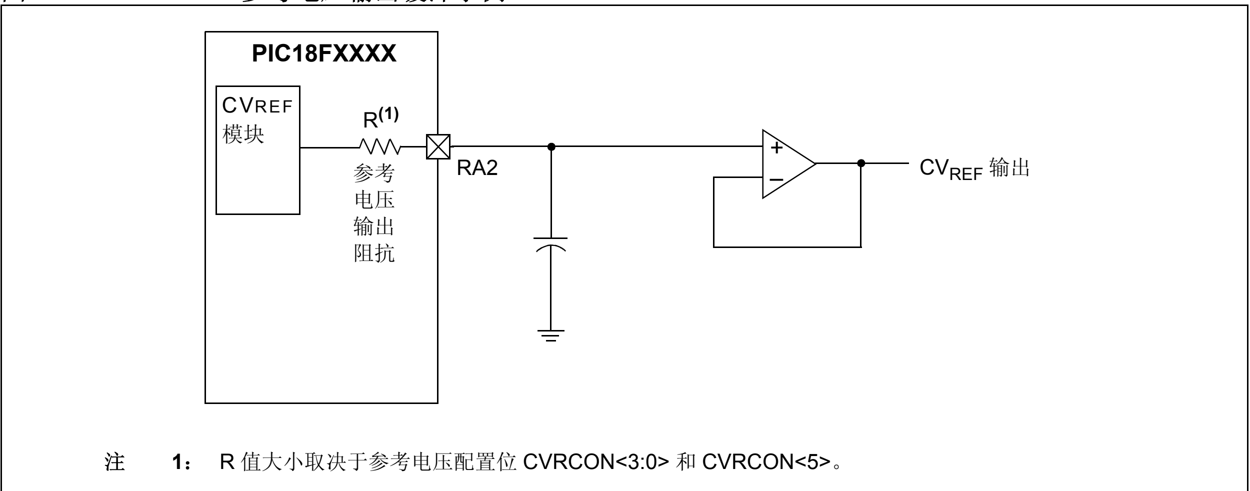
23.5 连接注意事项

参考电压模块的工作独立于比较器模块。如果 TRISA<2> 位和 CVROE 位都被置 1，参考电压的输出将被连接到 RA2 引脚。参考电压输出与 RA2 引脚相连时，如果该引脚被配置为数字输入引脚，将会增大电流消耗。使能 CVRSS 时将 RA2 用作数字输出引脚也将增加电流消耗。

RA2 引脚可被直接用作 D/A 输出，但是其驱动能力有限。由于这种有限的电流驱动能力，当参考电压输出模块连接外部 VREF 时必须使用缓冲器。图 23-2 举例说明了这一缓冲技术。

 初稿

图 23-2： 参考电压输出缓冲示例



注 1： R 值大小取决于参考电压配置位 CVRCON<3:0> 和 CVRCON<5>。

表 23-1： 与比较器参考电压模块相关的寄存器

名称	Bit 7	Bit 6	Bit 5	Bit 4	Bit 3	Bit 2	Bit 1	Bit 0	复位值所在页
CVRCON	CVREN	CVROE	CVRR	CVRSS	CVR3	CVR2	CVR1	CVR0	53
CMCON	C2OUT	C1OUT	C2INV	C1INV	CIS	CM2	CM1	CM0	53
TRISA	—	TRISA6[1]	TRISA5	TRISA4	TRISA3	TRISA2	TRISA1	TRISA0	54

图注： 比较器参考电压模块不使用阴影单元。

注 1： 根据振荡器配置使能 PORTA 引脚。

注：

24.0 高 / 低压检测 （HLVD）

PIC18F2455/2550/4455/4550 器件有一个高 / 低压检测模块 （HLVD）。该模块是一个可编程电路，它允许用户指定器件的电压跳变点和相对于该点的电压变化方向。如果器件电压按照指定的方向相对于该跳变点发生了偏移，就会将中断标志位置 1。如果允许了中断，程序就会转移到中断向量地址处执行，由软件响应该中断。

高 / 低压检测控制寄存器（寄存器 24-1）控制 HLVD 模块的工作。这就允许用户用软件控制使该电路“关闭”，从而使器件的电流消耗降至最低。

图 24-1 所示为 HLVD 模块的框图。

寄存器 24-1： HLVDCON：高 / 低压检测控制寄存器

R/W-0	U-0	R-0	R/W-0	R/W-0	R/W-1	R/W-0	R/W-1
VDIRMAG	—	IRVST	HLVDEN	HLVDL3	HLVDL2	HLVDL1	HLVDL0
bit 7							bit 0

bit 7 **VDIRMAG**：电压方向选择位

1 = 当电压等于或超过跳变点 （HLVDL3:HLVDL0）时发生事件。
0 = 当电压等于或小于跳变点 （HLVDL3:HLVDL0）时发生事件。

bit 6 未用：读为 0

bit 5 **IRVST**：内部参考电压稳定标志位

1 = 表示电压检测逻辑将在指定的电压范围产生中断标志
0 = 表示电压检测逻辑将不会在指定的电压范围产生中断标志，并且不允许 HLVD 中断

bit 4 **HLVDEN**：高 / 低压检测电源使能位

1 = 使能 HLVD
0 = 禁止 HLVD

bit 3-0 **HLVDL3:HLVDL0**：电压检测门限位 (1)

1111 = 使用外部模拟输入 （输入来自于 HLVDIN 引脚）
1110 = 最大设置
.
.
.
0000 = 最小设置

注 1：相应规范请参见**第 28.0 节 “电气规范”**中的表 28-6。

图注：			
R = 可读位	W = 可写位	U = 未用位，读为 0	
-n = 上电复位时的值	1 = 置 1	0 = 清零	x = 未知

通过将 HLVDEN 位置 1 使能该模块。每次使能 HLVD 模块时，电路需要一定延时才能稳定下来。IRVST 位是一个只读位，用来表明电路是否稳定。仅当该电路稳定且 IRVST 位置 1 时，该模块才能产生中断。

VDIRMAG 位决定模块的整体工作状态。当 VDIRMAG 清零时，模块监视 VDD 是否跌落到预先确定的设置点以下。当该位置 1 时，模块监视 VDD 是否上升到设置点以上。

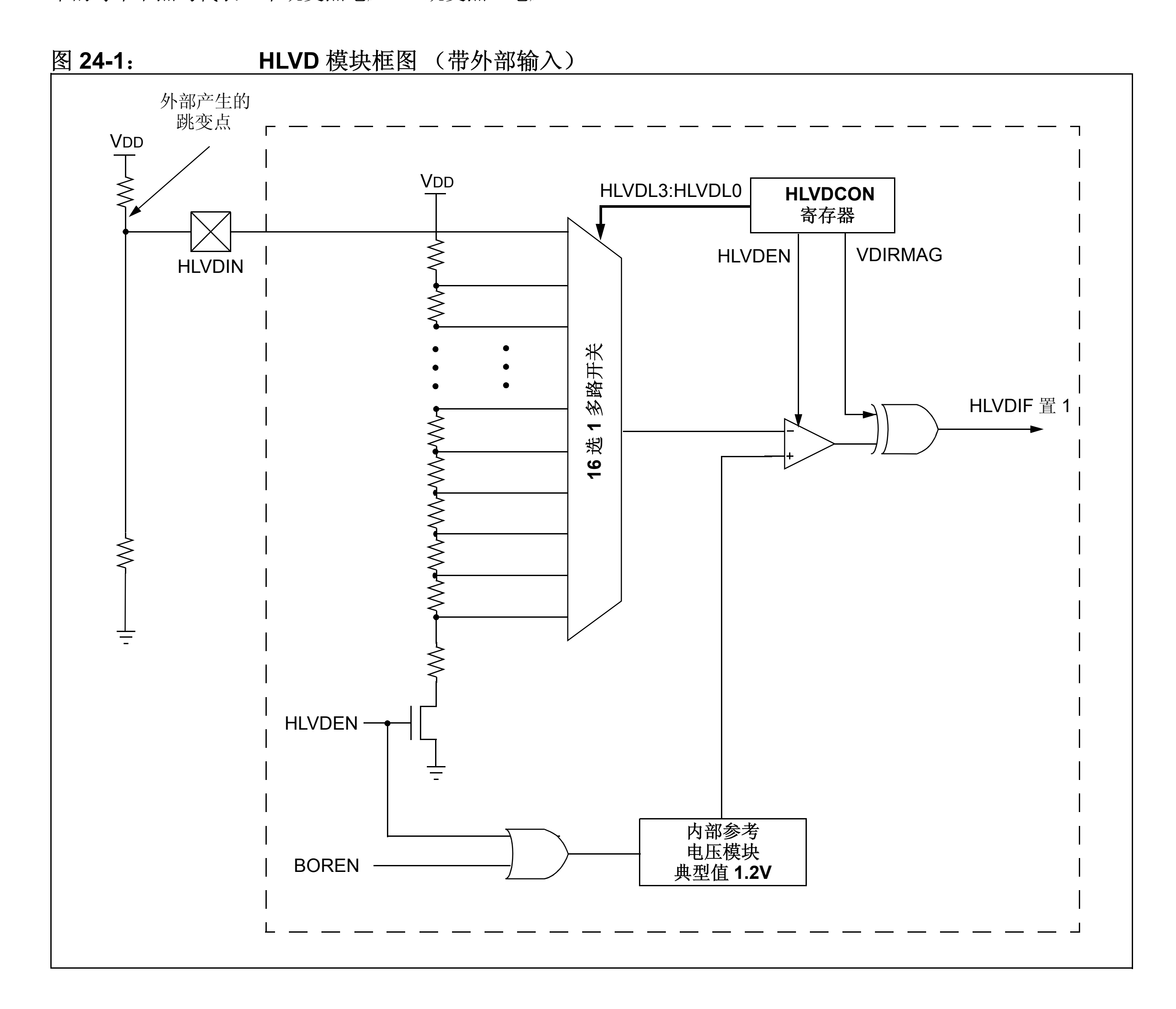
24.1 工作原理

当使能 HLVD 模块时，比较器使用内部产生的参考电压作为设置点。设置点与跳变点作比较，其中电阻分压器中的每个节点均代表一个跳变点电压。“跳变点”电压是器件在检测到高压或低压事件时的电平，它取决于该模块的配置。当供电电压等于跳变点电压时，电阻阵列分压值等于由参考电压模块产生的内部参考电压。然后比较器通过将 HLVDIF 位置 1 产生一个中断信号。

可用软件编程指定跳变点电压为 16 个值中的任何一个。通过对 HLVDL3:HLVDL0 位（HLVDCON<3:0>）进行编程可以选择跳变点。

HLVD 模块还有一个功能，可以让用户通过外部源向模块提供跳变电压。当 HLVDL3:HLVDL0 位被置为 1111 时，使能此模式。在此状态下，比较器输入与外部输入引脚 HLVDIN 复用。因此用户可以灵活地进行配置，以在有效工作范围内的任何电压产生高 / 低压检测中断。

图 24-1： HLVD 模块框图 （带外部输入）



 初稿

24.2 设置 HLVD

要设置 HLVD 模块，需要遵循以下步骤：

1. 将 HLVDEN 位（HLVDCON<4>）清零禁止模块。
2. 将值写入 HLVDL3:HLVDL0 位，选择所需的 HLVD 跳变点。
3. 将 VDIRMAG 位设置为检测高于跳变点电压（VDIRMAG = 1）或低于跳变点电压（VDIRMAG = 0）。
4. 通过将 HLVDEN 位置 1 使能 HLVD 模块。
5. 将 HLVD 中断标志位 HLVIF （PIR2<2>）清零，该位可能从上次中断起一直保持置 1。
6. 如果需要中断，通过将 HLVDIE 和 GIE/GIEH 位置 1 （PIE2<2> 和 INTCON<7>）允许 HLVD 中断。直到 IRVST 位置 1 后才会发生中断。

24.3 电流消耗

使能了该模块就使能了 HLVD 比较器和分压器，并将消耗静态电流。电气规范中的参数 D022（第 **28.2 节“直流特性”**）给出了使能该模块时的电流总消耗。

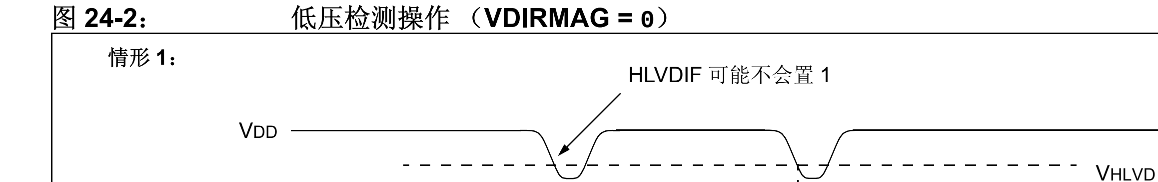
HLVD 模块不需要一直工作，工作与否取决于具体的应用。要降低电流消耗，只需要在检测电压时，短时间地使能 HLVD 电路，而在检测完成之后马上禁止 HLVD 模块。

24.4 HLVD 启动时间

电气规范中的参数 D420（见第 **28.0 节“电气规范”**中的表 28-6）给出了 HLVD 模块的内部参考电压，该参考电压也可由其他内部电路使用，例如可编程欠压复位电路。如果禁止了 HLVD 或其他使用参考电压的电路以降低器件的电流消耗，则参考电压电路将需要一段时间稳定下来以后才能可靠地检测低压或高压条件。 HLVD 启动时间 TIRVST 是一个独立于器件时钟速率的时间间隔，由电气规范参数 36 指定 （见表 28-12）。

直到 TIRVST 结束并且参考电压达到稳定后，才会允许 HLVD 中断。由于这个原因，在此时间间隔期间，略微超出设置点的偏移可能不会被检测到。请参见图 24-2 或图 24-3。

图 24-2： 低压检测操作 （VDIRMAG = 0）



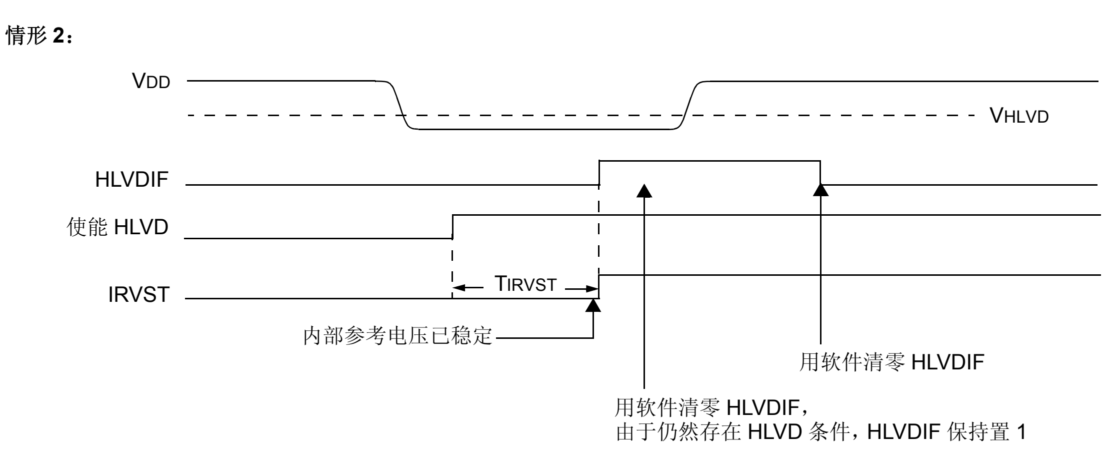


图 24-3： 高压检测操作 （VDIRMAG = 1）

情形 1：
HLVDIF 可能没有置 1
VHLVD
VDD
HLVDIF
使能 HLVD
IRVST
TIRVST
用软件清零 HLVDIF
内部参考电压已稳定

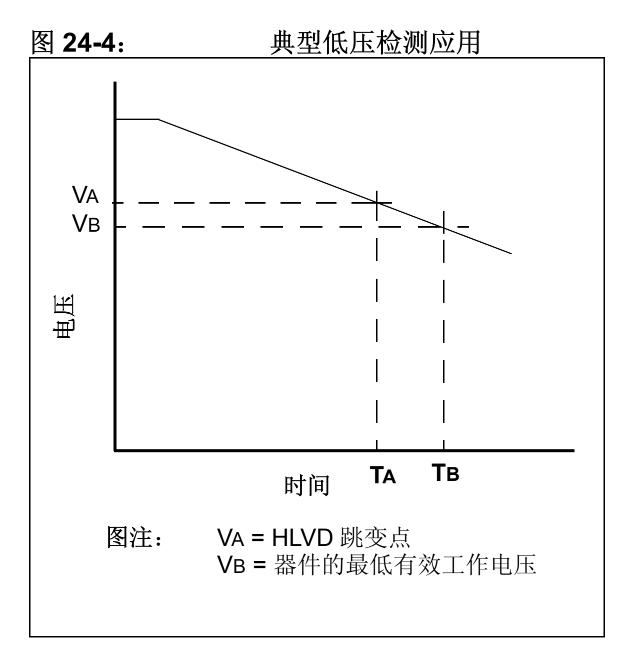
情形 2：
VHLVD
VDD
HLVDIF
使能 HLVD
IRVST
TIRVST
内部参考电压已稳定
用软件清零 HLVDIF
用软件清零 HLVDIF，
由于仍然存在 HLVD 条件，HLVDIF 保持置 1

24.5 应用

在许多应用中，需要有检测电压跌落到某个特定门限值以下或以上的功能。例如，可以定期使能 HLVD 模块来检测是否连接 USB（Universal Serial Bus）。这里假定器件的供电电压低于未接外设的 USB 电压。若连接了 USB，将检测到高压条件（假设 USB 上的电压为 3.3V 到 5V）；如果断开连接，情况正好相反。该功能可以省去设计一些额外元件和连接信号 （输入引脚）。

对于一般的电池应用，图 24-4 显示了一个可能的电压曲线。器件电压会随时间逐渐下降。当器件电压达到电压 V_A 时， HLVD 逻辑电路会在 T_A 时产生中断。中断会导致执行 ISR，从而使应用程序能在器件电压退出有效工作范围 （对应时间为 T_B）之前执行 “日常任务”并安全关闭。因此， HLVD 将会给该应用提供一个时间窗（表示为 T_A 和 T_B 的差）使应用程序能安全地退出。

图 24-4： 典型低压检测应用



24.6 休眠期间的操作

使能时， HLVD 电路在休眠期间将继续工作。如果器件电压越过了跳变点， HLVDIF 位将会被置 1，并且将唤醒器件。如果已经允许了全局中断，程序将从中断向量地址处继续执行。

24.7 复位的影响

器件复位强制所有寄存器进入复位状态。这会强制关闭 HLVD 模块。

表 24-1： 与高 / 低压检测模块相关的寄存器

名称	Bit 7	Bit 6	Bit 5	Bit 4	Bit 3	Bit 2	Bit 1	Bit 0	复位值所在页
HLVDCON	VDIRMAG	—	IRVST	HLVDEN	HLVDL3	HLVDL2	HLVDL1	HLVDL0	52
INTCON	GIE/GIEH	PEIE/GIEL	TMR0IE	INT0IE	RBIE	TMR0IF	INT0IF	RBIF	51
PIR2	OSCFIF	CMIF	USBIF	EEIF	BCLIF	HLVDIF	TMR3IF	CCP2IF	54
PIE2	OSCFIE	CMIE	USBIE	EEIE	BCLIE	HLVDIE	TMR3IE	CCP2IE	54
IPR2	OSCFIP	CMIP	USBIP	EEIP	BCLIP	HLVDIP	TMR3IP	CCP2IP	54

图注： — = 未用位 （读为 0）。 HLVD 模块不使用阴影单元。

注：

25.0 CPU 的特殊性能

PIC18F2455/2550/4455/4550 器件包含的功能旨在最大限度地提高系统可靠性，并通过减少外部元件把系统成本降到最低。这些功能包括：

- 振荡器选择
- 复位：
  - 上电复位 （POR）
  - 上电延时定时器 （PWRT）
  - 振荡器起振定时器 （OST）
  - 欠压复位 （BOR）
- 中断
- 看门狗定时器 （WDT）
- 故障保护时钟监视器 （FSCM）
- 双速启动
- 代码保护
- ID 单元
- 在线串行编程

应根据具体应用对频率、功耗、精度和成本的要求配置振荡器。在**第 2.0 节“振荡器配置”**中详细讨论了所有的选项。

在本数据手册的前面几章中已经完整地讨论了器件的复位和中断。

除了为复位提供了上电延时定时器和振荡器起振定时器之外，PIC18F2455/2550/4455/4550 器件还提供了一个看门狗定时器，该定时器可通过配置位永久使能或由软件控制（如果配置为禁止）。

器件自带的 RC 振荡器还提供了故障保护时钟监视器（FSCM）和双速启动两个功能。 FSCM 对外设时钟进行后台监视，并在外设时钟发生故障时自动切换时钟源。双速启动使得代码几乎可在启动开始时立即执行，此时主时钟源正在完成自身的起振延时。

通过设置相应的配置寄存器位可以使能和配置所有这些功能。

25.1 配置位

可以通过对配置位编程 （读为 0）或不编程 （读为 1）来选择不同的器件配置。这些配置位被映射到程序存储器中从 300000h 开始的单元中。

用户会注意到地址 300000h 超出了用户程序存储空间的范围。事实上，它属于配置存储空间 （300000h — 3FFFFFh），该空间仅能通过表读和表写进行访问。

对配置寄存器编程类似于对闪存存储器编程。EECON1 寄存器中的 WR 位可启动对配置寄存器的自定时写操作。在正常工作模式下， TBLPTR 指向配置寄存器， TBLWT 指令设置要用于写操作的地址和数据。将 WR 位置 1 可启动对配置寄存器的长写操作。每次往配置寄存器写入 1 个字节。要写入或擦除一配置单元，可用 TBLWT 指令分别对该单元写入 1 或 0。关于闪存编程的更多详情，请参见第 **6.5 节 “写入闪存程序存储器”**。

表 25-1： 配置位和器件 ID

寄存器名称		Bit 7	Bit 6	Bit 5	Bit 4	Bit 3	Bit 2	Bit 1	Bit 0	缺省 / 未编程值
300000h	CONFIG1L	—	—	USBDIV	CPUDIV1	CPUDIV0	PLLDIV2	PLLDIV1	PLLDIV0	--00 0000
300001h	CONFIG1H	IESO	FCMEN	—	—	FOSC3	FOSC2	FOSC1	FOSC0	00-- 0101
300002h	CONFIG2L	—	—	VREGEN	BORV1	BORV0	BOREN1	BOREN0	$\overline{\text{PWRTEN}}$	--01 1111
300003h	CONFIG2H	—	—	—	WDTPS3	WDTPS2	WDTPS1	WDTPS0	WDTEN	---1 1111
300005h	CONFIG3H	MCLRE	—	—	—	—	LPT1OSC	PBADEN	CCP2MX	1--- -011
300006h	CONFIG4L	$\overline{\text{DEBUG}}$	XINST	ICPRT	—	—	LVP	—	STVREN	100- -1-1
300008h	CONFIG5L	—	—	—	—	CP3[(1)]	CP2	CP1	CP0	---- 1111
300009h	CONFIG5H	CPD	CPB	—	—	—	—	—	—	11-- ----
30000Ah	CONFIG6L	—	—	—	—	WRT3[(1)]	WRT2	WRT1	WRT0	---- 1111
30000Bh	CONFIG6H	WRTD	WRTB	WRTC	—	—	—	—	—	111- ----
30000Ch	CONFIG7L	—	—	—	—	EBTR3[(1)]	EBTR2	EBTR1	EBTR0	---- 1111
30000Dh	CONFIG7H	—	EBTRB	—	—	—	—	—	—	-1-- ----
3FFFFEh	DEVID1	DEV2	DEV1	DEV0	REV4	REV3	REV2	REV1	REV0	xxxx xxxx[(2)]
3FFFFFh	DEVID2	DEV10	DEV9	DEV8	DEV7	DEV6	DEV5	DEV4	DEV3	0001 0010[(2)]

图注： x = 未知， u = 不变， — = 未用。阴影单元未用，读为 0。

注 **1：** 在 PIC18FX455 器件中不可用；保持该位置 1。

2： 关于 DEVID1 的值，请参见寄存器 25-15。 DEVID 寄存器为只读寄存器，用户不能对其进行编程。

寄存器 25-1： CONFIG1L：配置寄存器 1 的低字节 （字节地址为 300000h）

U-0	U-0	R/P-0	R/P-0	R/P-0	R/P-0	R/P-0	R/P-0
—	—	USBDIV	CPUDIV1	CPUDIV0	PLLDIV2	PLLDIV1	PLLDIV0
bit 7							bit 0

bit 7-6 未用：读为 0

bit 5 **USBDIV**：USB 时钟选择位 （仅用于全速 USB 模式； UCFG:FSEN = 1）
1 = USB 时钟源为 96 MHz PLL 的 2 分频信号
0 = USB 时钟源直接来自无后分频的主振荡器模块

bit 4-3 **CPUDIV1:CPUDIV0**：系统时钟后分频比选择位
对于 XT、 HS、 EC 和 ECIO 振荡器模式：
11 = 主振荡器 4 分频作为系统时钟
10 = 主振荡器 3 分频作为系统时钟
10 = 主振荡器 2 分频作为系统时钟
00 = 主振荡器直接用作系统时钟 （无后分频器）
对于 XTPLL、 HSPLL、 ECPLL 和 ECPIO 振荡器模式：
11 = 96 MHz PLL 6 分频作为系统时钟
10 = 96 MHz PLL 4 分频作为系统时钟
11 = 96 MHz PLL 3 分频作为系统时钟
10 = 96 MHz PLL 2 分频作为系统时钟

bit 2-0 **PLLDIV2:PLLDIV0**：PLL 预分频比选择位
111 = 12 分频 （48 MHz 振荡器输入）
110 = 10 分频 （40 MHz 振荡器输入）
101 = 6 分频 （24 MHz 振荡器输入）
100 = 5 分频 （20 MHz 振荡器输入）
011 = 4 分频 （16 MHz 振荡器输入）
010 = 3 分频 （12 MHz 振荡器输入）
001 = 2 分频 （8 MHz 振荡器输入）
000 = 不分频 （4 MHz 振荡器输入直接驱动 PLL）

图注：			
R = 可读位	P= 可编程位	U = 未用位，读为 0	
-n = 上电复位时的值	1 = 置 1	0 = 清零	x = 未知

寄存器 25-2： **CONFIG1H：配置寄存器 1 的高字节 （字节地址为 300001h）**

R/P-0	R/P-0	U-0	U-0	R/P-0	R/P-1	R/P-0	R/P-1
IESO	FCMEN	—	—	FOSC3	FOSC2	FOSC1	FOSC0
bit 7							bit 0

bit 7 **IESO：**内部 / 外部振荡器切换位
1 = 使能振荡器切换模式
0 = 禁止振荡器切换模式

bit 6 **FCMEN：**故障保护时钟监视器使能位
1 = 使能故障保护时钟监视器
0 = 禁止故障保护时钟监视器

bit 5-4 未用：读为 0

bit 3-0 **FOSC3:FOSC0：**振荡器选择位 [(1)]
111x = HS 振荡器，使能 PLL （HSPLL）
110x = HS 振荡器 （HS）
1011 = 内部振荡器， USB 使用 HS 振荡器 （INTHS）
1010 = 内部振荡器， USB 使用 XT 振荡器 （INTXT）
1001 = 内部振荡器， RA6 用作 CLKO， USB 使用 EC 振荡器 （INTCKO）
1000 = 内部振荡器， RA6 用作端口， USB 使用 EC 振荡器 （INTIO）
0111 = EC 振荡器，使能 PLL， RA6 用作 CLKO （ECPLL）
0110 = EC 振荡器，使能 PLL， RA6 用作端口 （ECPIO）
0101 = EC 振荡器， RA6 用作 CLKO （EC）
0100 = EC 振荡器， RA6 用作端口 （ECIO）
001x = XT 振荡器，使能 PLL （XTPLL）
000x = XT 振荡器 （XT）

注 **1：**在 XT、HS 和 EC 模式下，单片机和 USB 模块使用同一振荡器作为时钟源。当单片机使用内部振荡器模块时， USB 模块仍使用选定的 XT、 HS 或 EC 振荡器。

图注：		
R = 可读位	P = 可编程位	U = 未用位，读为 0
-n = 未对器件编程时的值		u = 编程后状态不变

寄存器 25-3： **CONFIG2L：配置寄存器 2 的低字节 （字节地址 300002h）**

U-0	U-0	R/P-0	R/P-1	R/P-1	R/P-1	R/P-1	R/P-1
—	—	VREGEN	BORV1[1]	BORV0[1]	BOREN1[2]	BOREN0[2]	$\overline{\text{PWRTEN}}$[2]
bit 7							bit 0

bit 7-6 未用：读为 0

bit 5 **VREGEN**：USB 内部稳压器使能位
1 = 使能 USB 稳压器
0 = 禁止 USB 稳压器

bit 4-3 **BORV1:BORV0**：欠压复位电压位 [1]
11 = 最小设置
.
.
.
00 = 最大设置

bit 2-1 **BOREN1:BOREN0**：欠压复位使能位 [2]
11 = 仅硬件可使能欠压复位 （禁止 SBOREN）
10 = 仅硬件可使能欠压复位，休眠模式下被禁止 （禁止 SBOREN）
01 = 由软件使能和控制欠压复位 （使能 SBOREN）
00 = 禁止使用硬件或软件控制欠压复位

bit 0 **$\overline{\text{PWRTEN}}$**：上电延时定时器使能位 [2]
1 = 禁止 PWRT
0 = 使能 PWRT

注 1：请参见第 **28.0 节 "电气规范"** 了解相关的规范。
2：上电延时定时器与欠压复位是相互分离的，这样可以分别控制两者的工作。

图注：		
R = 可读位	P = 可编程位	U = 未用位，读为 0
-n = 未对器件编程时的值		u = 编程后状态不变

寄存器 25-4： CONFIG2H：配置寄存器 2 的高字节 （字节地址为 300003h）

U-0	U-0	U-0	R/P-1	R/P-1	R/P-1	R/P-1	R/P-1
—	—	—	WDTPS3	WDTPS2	WDTPS1	WDTPS0	WDTEN
bit 7							bit 0

bit 7-5 未用：读为 0

bit 4-1 **WDTPS3:WDTPS0**：看门狗定时器后分频比选择位
1111 = 1:32,768
1110 = 1:16,384
1101 = 1:8,192
1100 = 1:4,096
1011 = 1:2,048
1010 = 1:1,024
1001 = 1:512
1000 = 1:256
0111 = 1:128
0110 = 1:64
0101 = 1:32
0100 = 1:16
0011 = 1:8
0010 = 1:4
0001 = 1:2
0000 = 1:1

bit 0 **WDTEN**：看门狗定时器使能位
1 = 使能 WDT
0 = 禁止 WDT （由 SWDTEN 位控制）

图注：		
R = 可读位	P = 可编程位	U = 未用位，读为 0
-n = 未对器件编程时的值		u = 编程后状态不变

寄存器 25-5： CONFIG3H：配置寄存器 3 的高字节 （字节地址 300005h）

R/P-1	U-0	U-0	U-0	U-0	R/P-0	R/P-1	R/P-1
MCLRE	—	—	—	—	LPT1OSC	PBADEN	CCP2MX
bit 7							bit 0

bit 7 **MCLRE**：MCLR 引脚使能位
1 = 使能 MCLR 引脚；禁止 RE3 输入引脚。
0 = 使能 RE3 引脚；禁止 MCLR 引脚

bit 6-3 未用：读为 0

bit 2 **LPT1OSC**：低功耗 Timer1 振荡器使能位
1 = Timer1 配置为低功耗运行
0 = Timer1 配置为高功耗运行

bit 1 **PBADEN**：PORTB A/D 使能位
（影响 ADCON1 复位状态。 ADCON1 控制 PORTB<4:0> 引脚配置。）
1 = 复位时 PORTB<4:0> 引脚配置为模拟输入通道
0 = 复位时 PORTB<4:0> 引脚配置为数字 I/O

bit 0 **CCP2MX**：CCP2 多路复用位
1 = CCP2 输入 / 输出与 RC1 复用
0 = CCP2 输入 / 输出与 RB3 复用

图注：		
R = 可读位	P = 可编程位	U = 未用位，读为 0
-n = 未对器件编程时的值		u = 编程后状态不变

寄存器 25-6： **CONFIG4L：配置寄存器 4 的低字节 （字节地址 300006h）**

R/P-1	R/P-0	R/P-0	U-0	U-0	R/P-1	U-0	R/P-1
$\overline{\text{DEBUG}}$	XINST	ICPRT[(1)]	—	—	LVP	—	STVREN
bit 7							bit 0

bit 7 **$\overline{\text{DEBUG}}$**：后台调试器使能位
1 = 禁止后台调试器， RB6 和 RB7 被配置为通用 I/O 引脚
0 = 使能后台调试器， RB6 和 RB7 专用于在线调试

bit 6 **XINST**：扩展指令集使能位
1 = 使能指令集扩展和变址寻址模式
0 = 禁止指令集扩展和变址寻址模式 （传统模式）

bit 5 **ICPRT**：专用在线调试 / 编程端口 （ICPORT）使能位 [(1)]
1 = 使能 ICPORT
0 = 禁止 ICPORT

注 **1**：仅用于 44 引脚 TQFP 封装的 PIC18F4455/4550 器件。在其他器件中始终保持该位清零。

bit 4-3 未用：读为 0

bit 2 **LVP**：单电源 ICSP™ 使能位
1 = 使能单电源 ICSP
0 = 禁止单电源 ICSP

bit 1 未用：读为 0

bit 0 **STVREN**：堆栈满 / 下溢复位使能位
1 = 堆栈满 / 下溢导致复位
0 = 堆栈满 / 下溢不导致复位

图注：		
R = 可读位	C= 可清零位	U = 未用位，读为 0
-n = 未对器件编程时的值		u = 编程后状态不变

寄存器 25-7： CONFIG5L：配置寄存器 5 的低字节 （字节地址 300008h）

U-0	U-0	U-0	U-0	R/C-1	R/C-1	R/C-1	R/C-1
—	—	—	—	CP3[(1)]	CP2	CP1	CP0
bit 7							bit 0

bit 7-4 未用：读为 0

bit 3 **CP3**：代码保护位 [(1)]
1 = 程序存储区 3 （006000 － 007FFFh）无代码保护
0 = 程序存储区 3 （006000 － 007FFFh）有代码保护

注 **1**：在 PIC18FX455 器件中不可用；保持该位置 1。

bit 2 **CP0**：代码保护位
1 = 程序存储区 2 （004000 － 005FFFh）无代码保护
0 = 程序存储区 2 （004000 － 005FFFh）有代码保护

bit 1 **CP1**：代码保护位
1 = 程序存储区 1 （002000 － 003FFFh）无代码保护
0 = 程序存储区 1 （002000 － 003FFFh）有代码保护

bit 0 **CP0**：代码保护位
1 = 程序存储区 0 （000800 － 000FFFh）无代码保护
0 = 程序存储区 0 （000800 － 000FFFh）有代码保护

图注：		
R = 可读位	C = 可清零位	U = 未用位，读为 0
-n = 未对器件编程时的值		u = 编程后状态不变

寄存器 25-8： CONFIG5H：配置寄存器 5 的高字节 （字节地址 300009h）

R/C-1	R/C-1	U-0	U-0	U-0	U-0	U-0	U-0
CPD	CPB	—	—	—	—	—	—
bit 7							bit 0

bit 7 **CPD**：数据 EEPROM 代码保护位
1 = 数据 EEPROM 无代码保护
0 = 数据 EEPROM 有代码保护

bit 6 **CPB**：引导区代码保护位
1 = 引导区 （000000-0007FFh）无代码保护
0 = 引导区 （000000-0007FFh）有代码保护

bit 5-0 未用：读为 0

图注：		
R = 可读位	C = 可清零位	U = 未用位，读为 0
-n = 未对器件编程时的值		u = 编程后状态不变

寄存器 25-9： CONFIG6L：配置寄存器 6 的低字节 （字节地址 30000Ah）

U-0	U-0	U-0	U-0	R/C-1	R/C-1	R/C-1	R/C-1
—	—	—	—	WRT3[1]	WRT2	WRT1	WRT0
bit 7							bit 0

bit 7-4 未用：读为 0

bit 3 **WRT3**：写保护位 [1]
1 = 程序存储区 3 （006000 － 007FFFh）无写保护
0 = 程序存储区 3 （006000 － 007FFFh）有写保护

注 **1**：在 PIC18FX455 器件中不可用；保持该位置 1。

bit 2 **WRT2**：写保护位
1 = 程序存储区 2 （004000 － 005FFFh）无写保护
0 = 程序存储区 2 （004000 － 005FFFh）有写保护

bit 1 **WRT1**：写保护位
1 = 程序存储区 1 （002000 － 003FFFh）无写保护
0 = 程序存储区 1 （002000 － 003FFFh）有写保护

bit 0 **WRT0**：写保护位
1 = 程序存储区 0 （000800-001FFFh）无写保护
0 = 程序存储区 0 （000800-001FFFh）有写保护

图注：		
R = 可读位	C = 可清零位	U = 未用位，读为 0
-n = 未对器件编程时的值		u = 编程后状态不变

寄存器 25-10： CONFIG6H：配置寄存器 6 的高字节 （字节地址 30000Bh）

R/C-1	R/C-1	R-1	U-0	U-0	U-0	U-0	U-0
WRTD	WRTB	WRTC[1]	—	—	—	—	—
bit 7							bit 0

bit 7 **WRTD**：数据 EEPROM 写保护位
1 = 数据 EEPROM 无写保护
0 = 数据 EEPROM 有写保护

bit 6 **WRTB**：引导区写保护位
1 = 引导区 （000000-0007FFh）无写保护
0 = 引导区 （000000-0007FFh）有写保护

bit 5 **WRTC**：配置寄存器写保护位 [1]
1 = 配置寄存器 （300000-3000FFh）无写保护
0 = 配置寄存器 （300000-3000FFh）有写保护

注 **1**：在正常执行模式下，该位是只读的；仅在编程模式下才可写入该位。

bit 4-0 未用：读为 0

图注：		
R = 可读位	C = 可清零位	U = 未用位，读为 0
-n = 未对器件编程时的值		u = 编程后状态不变

寄存器 25-11： CONFIG7：配置寄存器 7 的低字节 （字节地址 30000Ch）

U-0	U-0	U-0	U-0	R/C-1	R/C-1	R/C-1	R/C-1
—	—	—	—	EBTR3[(1)]	EBTR2	EBTR1	EBTR0
bit 7							bit 0

bit 7-4 未用：读为 0

bit 3 **EBTR3**：表读保护位 [(1)]

1 = 程序存储区 3 （006000-007FFFh）无表读保护，可从其他存储区对其执行表读操作
0 = 程序存储区 3 （006000-007FFFh）有表读保护，不能从其他区块对其执行表读操作

注 **1**：在 PIC18FX455 器件中不可用；保持该位置 1。

bit 2 **EBTR2**：表读保护位

1 = 程序存储区 2 （004000-005FFFh）无表读保护，可从其他区块对其执行表读操作
0 = 程序存储区 2 （004000-005FFFh）有表读保护，不能从其他区块对其执行表读操作

bit 1 **EBTR1**：表读保护位

1 = 程序存储区 1 （002000-003FFFh）无表读保护，可从其他区块对其执行表读操作
0 = 程序存储区 1 （002000-003FFFh）有表读保护，不能从其他区块对其执行表读操作

bit 0 **EBTR0**：表读保护位

1 = 程序存储区 0 （000800-001FFFh）无表读保护，可从其他区块对其执行表读操作
0 = 程序存储区 0 （000800-001FFFh）有表读保护，不能从其他区块对其执行表读操作

图注：		
R = 可读位	C = 可清零位	U = 未用位，读为 0
-n = 未对器件编程时的值		u = 编程后状态不变

寄存器 25-12： CONFIG7H：配置寄存器 7 的高字节 （字节地址 30000Dh）

U-0	R/C-1	U-0	U-0	U-0	U-0	U-0	U-0
—	EBTRB	—	—	—	—	—	—
bit 7							bit 0

bit 7 未用：读为 0

bit 6 **EBTRB**：引导区表读保护位

1 = 引导区 （000000-0007FFh）无表读保护，可从其他区块对其执行表读操作
0 = 引导区 （000000-0007FFh）有表读保护，不能从其他区块对其执行表读操作

bit 5-0 未用：读为 0

图注：		
R = 可读位	C = 可清零位	U = 未用位，读为 0
-n = 未对器件编程时的值		u = 编程后状态不变

寄存器 25-13： DEVID1：PIC18F2455/2550/4455/4550 的器件 ID 寄存器 1

R	R	R	R	R	R	R	R
DEV2	DEV1	DEV0	REV4	REV3	REV2	REV1	REV0
bit 7							bit 0

bit 7-5 **DEV2:DEV0**：器件 ID 位
`011` = PIC18F2455
`010` = PIC18F2550
`001` = PIC18F4455
`000` = PIC18F4550

bit 4-0 **REV4:REV0**：版本 ID 位
这些位用于标识器件版本。

图注：		
R = 只读位	P = 可编程位	U = 未用位，读为 0
-n = 未对器件编程时的值		u = 编程后状态不变

寄存器 25-14： DEVID2：PIC18F2455/2550/4455/4550 的器件 ID 寄存器 2

R	R	R	R	R	R	R	R
DEV10	DEV9	DEV8	DEV7	DEV6	DEV5	DEV4	DEV3
bit 7							bit 0

bit 7-0 **DEV10:DEV3**：器件 ID 位
这些位与器件 ID 寄存器 1 中的 DEV2:DEV0 位一起用于标识器件编号。
`0001 0010` = PIC18F2455/2550/4455/4550 器件

注： DEV10:DEV3 的值可能会与其他器件共享使用。特定器件都是通过使用整个 DEV10:DEV0 位序列加以标识的。

图注：		
R = 只读位	P = 可编程位	U = 未用位，读为 0
-n = 未对器件编程时的值		u = 编程后状态不变

25.2 看门狗定时器（WDT）

PIC18F2455/2550/4455/4550 器件的 WDT 由 INTRC 时钟源驱动。当使能 WDT 时，也将同时使能时钟源。WDT 超时溢出周期的标称值为 4 ms，其稳定性与 INTRC 振荡器相当。

4 ms 的 WDT 周期与 16 位的后分频器中的值相乘。通过配置寄存器2H中的位控制一个多路开关以对WDT后分频器的输出进行选择。因此可获得范围为 4 ms 至 131.072 秒（2.18 分钟）的超时周期。当发生以下任一事件时，WDT 和后分频器将被清零，这些事件包括：执行 SLEEP 或 CLRWDT 指令、IRCF 位（OSCCON<6:4>）发生改变或发生时钟故障。

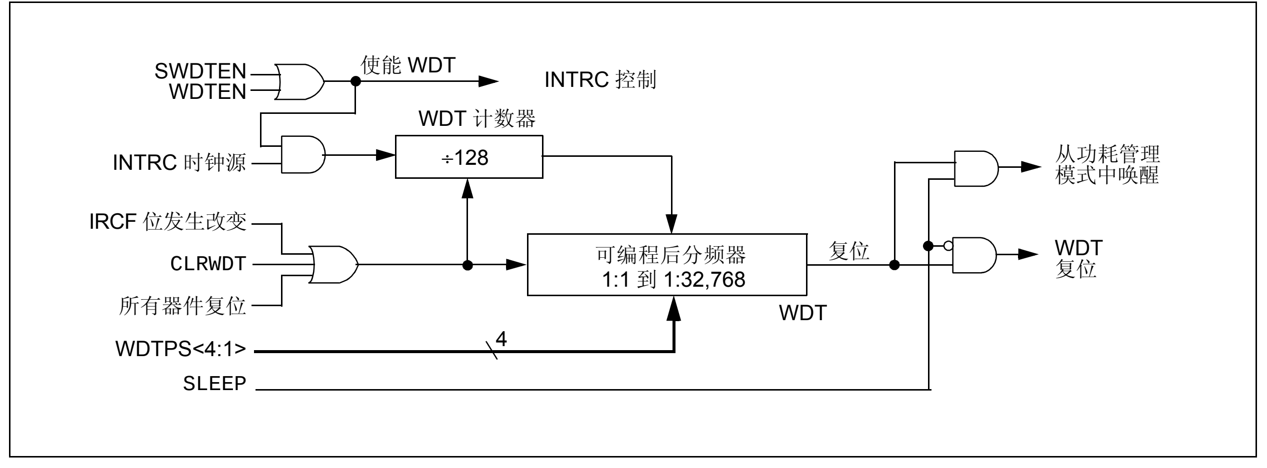
.

> 注 1： CLRWDT 和 SLEEP 指令会清零 WDT 和后分频器。
>
> 2： 更改 IRCF 位（OSCCON<6:4>）的设置会清零 WDT 和后分频器的计数值。
>
> 3： 当执行 CLRWDT 指令时，后分频器的计数值将被清零。

25.2.1 控制寄存器

寄存器 25-15 所示为 WDTCON 寄存器。这是一个可读写的寄存器，它包含一个控制位，仅当禁止 WDT 时，才能使用该控制位允许软件来控制 WDT。

图 25-1： WDT 框图



寄存器 25-15： **WDTCON：看门狗定时器控制寄存器**

U-0	U-0	U-0	U-0	U-0	U-0	U-0	R/W-0
—	—	—	—	—	—	—	SWDTEN[1]
bit 7							bit 0

bit 7-1 未用：读为 0

bit 0 **SWDTEN：**由软件控制的看门狗定时器使能位 [1]

1 = 打开看门狗定时器

0 = 关闭看门狗定时器

注 **1：** 当使能 WDTEN 配置位时，该位不起作用。

图注：	
R = 可读位	W = 可写位
U = 未用位，读为 0	-n = 上电复位时的值

表 **25-2：** 看门狗定时器的寄存器汇总

名称	Bit 7	Bit 6	Bit 5	Bit 4	Bit 3	Bit 2	Bit 1	Bit 0	复位值所在页
RCON	IPEN	SBOREN[1]	—	$\overline{\text{RI}}$	$\overline{\text{TO}}$	$\overline{\text{PD}}$	$\overline{\text{POR}}$	$\overline{\text{BOR}}$	52
WDTCON	—	—	—	—	—	—	—	SWDTEN	52

图注： — = 未用 （读为 0）。看门狗定时器不使用阴影单元。

注 **1：** 只有当 BOREN<1:0> = 01 时， SBOREN 位才可用，否则该位读为 0。

 初稿

25.3 双速启动

双速启动功能允许单片机在主时钟源可用之前使用 INTRC 振荡器作为时钟源，从而帮助器件最大限度地缩短从振荡器起振到代码执行之间的延时。通过将 IESO 配置位置 1 可使能该功能。

仅当主振荡器模式为 XT、HS、XTPLL 或 HSPLL（基于晶振的模式）时才可使用双速启动。其他时钟源不需要 OST 起振延时。对于这些时钟源，应禁止双速启动。

当使能双速启动时，器件复位和从休眠模式唤醒都会使器件将自身配置为运行在以内部振荡器电路作为时钟源的方式下（在发生上电复位后，还将由上电延时定时器产生一段延时）。这样几乎可使代码在主振荡器起振、OST 运行的同时立即执行。一旦 OST 超时，器件就自动切换到 PRI_RUN 模式。

因为 OSCCON 寄存器在复位时会被清零，INTOSC（或后分频器）时钟源在复位发生后不能马上使用；而 INTRC 时钟源可以直接以基本频率使用。为了在唤醒器件时使用更快的时钟速率，可以选择 INTOSC 或后分频器时钟源以提供更快的时钟速率，这可以通过在复位发生后立即设置 IRCF2:IRCF0 实现。对于从休眠模式唤醒的情况，可以通过在进入休眠模式之前设置 IRCF2:IRCF0 来选择 INTOSC 或后分频器时钟源。

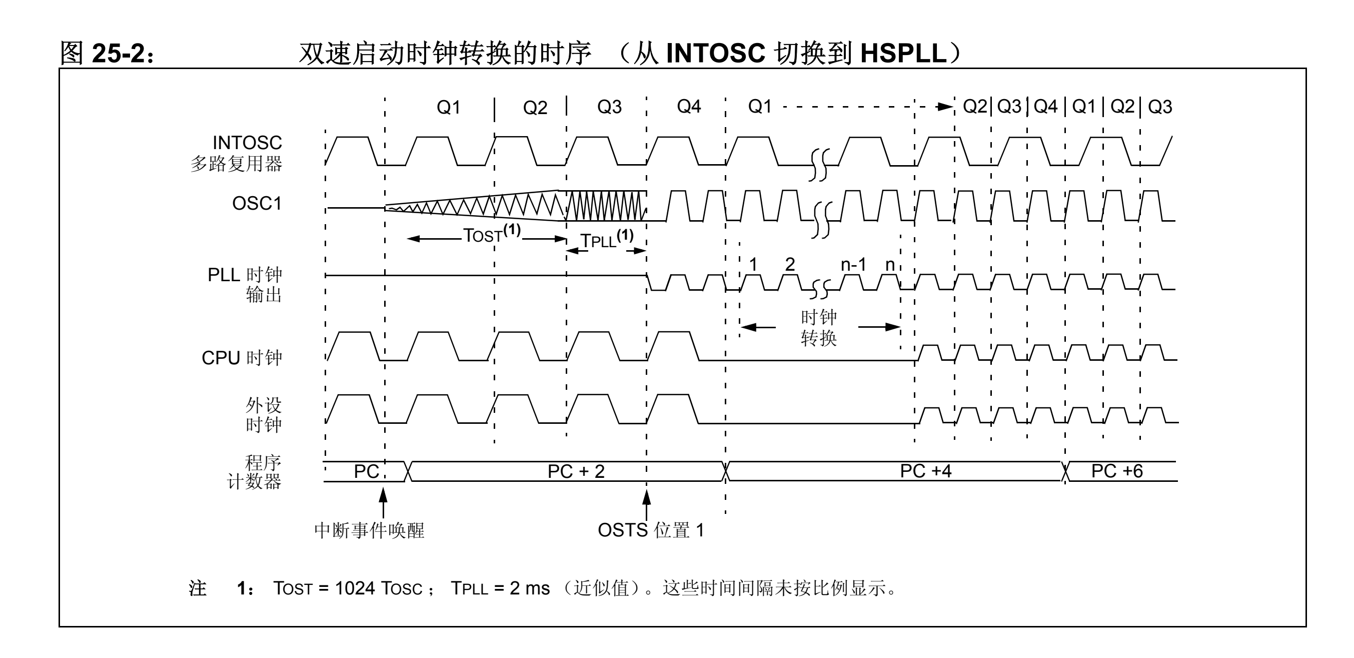
在其他的功耗管理模式下不使用双速启动。器件将使用当前选定的时钟源直到主时钟源可用为止。该操作与 IESO 位的设置无关。

25.3.1 使用双速启动时的注意事项

当在双速启动中使用 INTRC 振荡器时，器件仍将遵守进入功耗管理模式（包括执行 SLEEP 指令）的正常指令顺序（见第 **3.1.4 节 “多条 Sleep 命令”**）。实际上，这意味着在 OST 超时前用户代码可以改变 SCS1:SCS0 位的设置或执行 SLEEP 指令。这就使应用程序能短暂地唤醒器件，执行 “日常事务” 子程序，并在器件开始使用主时钟源前返回休眠状态。

用户代码还能通过查询 OSTS 位（OSCCON<3>）的状态来确定当前主时钟源是否正在为系统提供时钟。若该位置 1，则表示主振荡器正在为系统提供时钟。否则，表示当器件从复位或休眠模式唤醒期间由内部振荡器电路为系统提供时钟。

图 25-2： 双速启动时钟转换的时序 （从 **INTOSC** 切换到 **HSPLL**）



注 1： T_{OST} = 1024 T_{OSC}； T_{PLL} = 2 ms （近似值）。这些时间间隔未按比例显示。

25.4 故障保护时钟监视器

故障保护时钟监视器（Fail-Safe Clock Monitor，FSCM）可使单片机在发生外部时钟故障时，自动将系统时钟切换到内部振荡器电路以维持器件继续运行。将 FCMEN 配置位置 1 可使能 FSCM 功能。

当使能 FSCM 时， INTRC 振荡器将一直保持运行以监视外设时钟，并且在外设时钟发生故障时立即提供备用时钟。时钟监视 （如图 25-3 所示）通过创建一个采样时钟信号实现，该信号为 INTRC 输出的 64 分频。这样就使得 FSCM 采样时钟边沿之间有充足的时间间隔，从而保证在此间隔期间必然会出现外设时钟边沿。外设时钟和采样时钟作为时钟监视器锁存器 （CM）的输入。CM 在外设时钟的下降沿被置 1，在采样时钟的上升沿被清零。

图 25-3： FSCM 框图

时钟监视器
锁存器（CM）
（边沿触发）
外设
时钟
S Q
INTRC
时钟源
÷64
C Q̄
（32 µs）
488 Hz
（2.048 ms）
检测到
时钟故障

在采样时钟的下降沿检测外部时钟故障。如果在出现采样时钟的下降沿时， CM 仍置 1，就表示检测到外部时钟故障 （图 25-4）。这将引发以下事件：

- FSCM 将 OSCFIF （PIR2<7>）置 1，产生振荡器故障中断；
- 器件时钟源切换为内部振荡器电路 （OSCCON 不会被更新，因此无法显示当前时钟源——这就是故障保护状态）；并且
- WDT 复位。

切换过程中，对于时序要求较高的应用，内部振荡器电路的后分频频率可能不够稳定。在这些情况下，最好选择另一种时钟配置并进入其他功耗管理模式。可以尝试部分恢复或执行受控的关闭。请参见**第 3.1.4 节 “多条 Sleep 命令”**和**第 25.3.1 节“使用双速启动时的注意事项”**了解更多详细信息。

为了在唤醒器件时使用更快的时钟速率，可以选择 INTOSC 或后分频器时钟源以提供更快的时钟速率，这可以通过在复位发生后立即设置IRCF2:IRCF0实现。对于从休眠模式唤醒的情况，可以通过在进入休眠模式之前设置 IRCF2:IRCF0 来选择 INTOSC 或后分频器时钟源。

FSCM 只能检测出主时钟源或辅助时钟源的故障。如果内部振荡器电路发生故障，则无法检测到，当然也不会采取任何措施。

25.4.1 FSCM 和看门狗定时器

FSCM 和 WDT 均以 INTRC 振荡器作为时钟源。由于 WDT 使用独立的分频器和计数器，当使能 FSCM 时，禁止 WDT 对 INTRC 振荡器的运行没有影响。

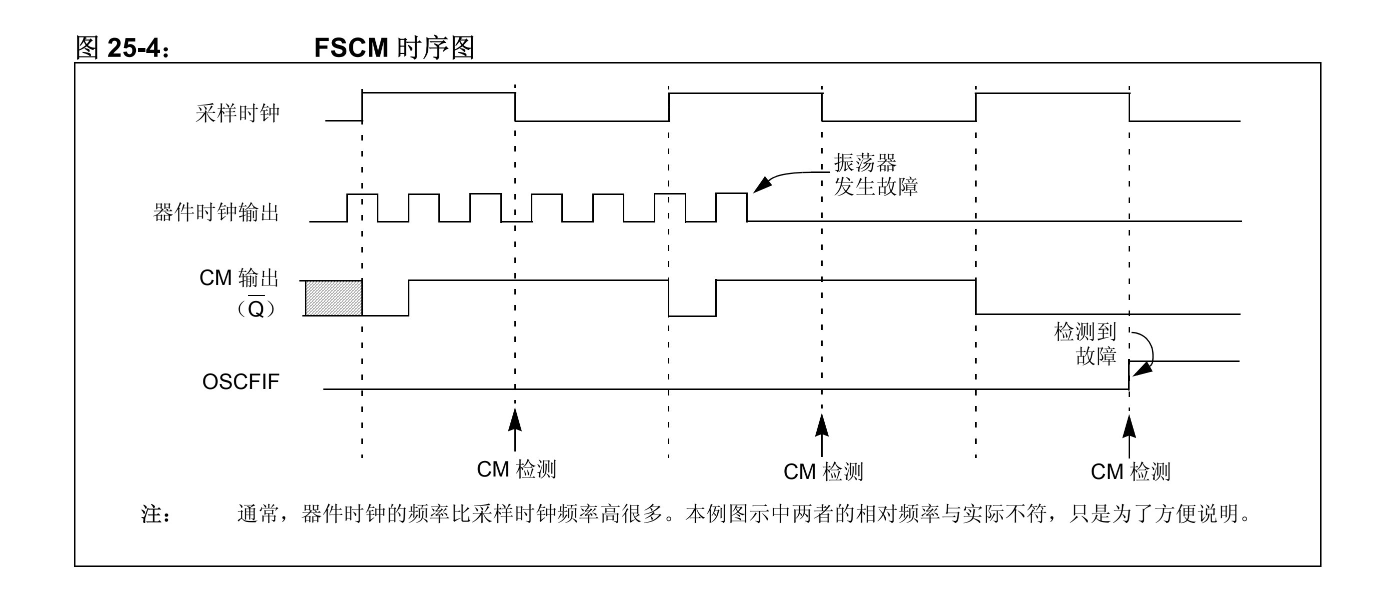
如前所述，当发生时钟故障时，时钟源将切换到 INTOSC 时钟源。根据 IRCF2:IRCF0 位选择的频率的不同，代码执行速度也会相应发生很大的变化。如果用小的预分频值使能 WDT，时钟速率的下降将导致 WDT 发生超时并在随后使器件复位。由于这个原因，故障保护事件也会将 WDT 和后分频器清零，使 WDT 从运行速率发生变化的时刻开始重新计数，从而减少发生错误超时的可能性。

25.4.2 退出故障保护运行模式

器件复位或进入功耗管理模式均可终止故障保护状态。发生复位时，控制器启动在配置寄存器 1H 中指定的主时钟源 （伴有如 OST 或 PLL 定时器等振荡器模式所需的起振延时）。 INTOSC 多路复用器将提供系统时钟直到主时钟源就绪为止 （类似于双速启动）。当主时钟源可用时，系统时钟源将切换回主时钟 （OSCCON 寄存器中的 OSTS 位置 1，表明当前使用的是主时钟源）。然后，故障保护时钟监视器恢复对外设时钟的监视。

在起振期间，主时钟源可能永远不能就绪。在这种情况下，器件将以 INTOSC 多路复用器作为时钟源。OSCCON 寄存器将保持复位状态直到进入功耗模式为止。

图 25-4： FSCM 时序图



注： 通常，器件时钟的频率比采样时钟频率高很多。本例图示中两者的相对频率与实际不符，只是为了方便说明。

25.4.3 功耗管理模式下的 FSCM 中断

进入功耗管理模式时，时钟多路开关选择由 OSCCON 寄存器选定的时钟源。在功耗管理模式下将恢复对功耗管理时钟源的故障保护监视。

如果在功耗管理模式下发生了振荡器故障，随后的操作将取决于是否允许振荡器故障中断。如果允许（OSCFIF = 1），代码执行将以 INTOSC 多路复用器作为时钟源，并且不会自动转回到发生故障的时钟源。

如果禁止该中断，空闲模式下振荡器故障所导致的中断将使 CPU 开始执行指令，此时由 INTOSC 时钟源为 CPU 提供时钟。

25.4.4 上电复位或从休眠中唤醒

FSCM 用于在器件退出上电复位（POR）或低功耗休眠模式后来检测振荡器故障。当系统主时钟为 EC 或 INTRC 模式时，对时钟源的监视会在这些事件发生后立即开始。

对于涉及到晶振或谐振器的振荡器模式（如 HS、HSPLL 或 XT），情况会有些不同。由于这类振荡器需要的起振时间可能比 FCSM 采样时钟周期长很多，因此可能会检测到假时钟故障。为了避免这一情况，内部振荡器电路会被自动配置为器件时钟直到主时钟稳定下来为止（OST 和 PLL 定时器已完成延时）。这与双速启动模式相同。一旦主时钟稳定下来，INTRC 就将重新作为 FSCM 时钟源。

> **注：** 用于防止上电复位或从休眠状态唤醒时发生假中断的逻辑电路，同样也将阻止在发生这些事件后对振荡器故障的检测。通过监视 OSTS 位，并使用定时程序来确定振荡器起振时间是否过长可避免这个问题。即便如此，在检测到振荡器故障时也不会将中断标志位置 1。

正如**第 25.3.1 节 "使用双速启动时的注意事项"**中所述，在等待主时钟稳定的过程中，可以选择另一种时钟配置并进入某一功耗管理模式。当选择新的功耗管理模式时，主时钟将被禁止。

25.5 程序校验和代码保护

PIC18 闪存器件的整个代码保护结构与其他的 PICmicro® 系列器件有很大的区别。

用户程序存储器被分为五个存储区。其中一个存储区为 2KB 的引导区，剩余部分被分为四个存储区。

每个存储区都有三个代码保护位与之相关联。它们是：

- 代码保护位 （CPn）
- 写保护位 （WRTn）
- 外部存储区表读位 （EBTRn）

图 25-5 显示了 24 KB 和 32 KB 器件的程序存储器构成，以及与每个存储区相关的特定代码保护位。表 25-3 总结了这些位的具体地址。

图 25-5： PIC18F2455/2550/4455/4550 的代码保护程序存储器

存储容量 / 器件

24 KB（PIC18F2455/2555）	32 KB（PIC18F2550/4550）	寻址范围	存储区保护代码：
引导区	引导区	000000h 0007FFh	CPB、 WRTB 和 EBTRB
存储区 0	存储区 0	000800h 001FFFh	CP0、 WRT0 和 EBTR0
存储区 1	存储区 1	002000h 003FFFh	CP1、 WRT1 和 EBTR1
存储区 2	存储区 2	004000h 005FFFh	CP2、 WRT2 和 EBTR2
未用 读为 0	存储区 3	006000h 007FFFh	CP3、 WRT3 和 EBTR3
未用 读为 0	未用 读为 0	008000h 1FFFFFh	（未用的存储空间）

表 25-3： 代码保护寄存器汇总

寄存器名称		Bit 7	Bit 6	Bit 5	Bit 4	Bit 3	Bit 2	Bit 1	Bit 0
300008h	CONFIG5L	—	—	—	—	CP3(1)	CP2	CP1	CP0
300009h	CONFIG5H	CPD	CPB	—	—	—	—	—	—
30000Ah	CONFIG6L	—	—	—	—	WRT3(1)	WRT2	WRT1	WRT0
30000Bh	CONFIG6H	WRTD	WRTB	WRTC	—	—	—	—	—
30000Ch	CONFIG7L	—	—	—	—	EBTR3(1)	EBTR2	EBTR1	EBTR0
30000Dh	CONFIG7H	—	EBTRB	—	—	—	—	—	—

图注： 未使用阴影单元。

注 1： 在 PIC18FX455 器件中不可用；保持该位置 1。

25.5.1 程序存储器代码保护

使用表读和表写指令可以对程序存储器的任何单元进行读写操作。器件 ID 可由表读指令读取。也可使用表读和表写指令对配置寄存器进行读写操作。

在正常执行模式下，CPn 位用于禁止来自外部的读写操作。如果 WRTn 配置位是 0，即可保护用户存储区不受表写指令的影响。而 EBTRn 位控制表读操作。如果用户某一存储区的 EBTRn 位为 0，允许在该存储区内部使用表读指令执行读操作，但不允许其他存储区对该存储区执行表读操作，否则读出的结果为 0。图 25-6 到图 25-8 说明了表读和表写保护的操作。

注：	代码保护位仅能从 1 状态改写为 0 状态。而不可能从 0 状态改写到 1 状态。只有使用整个芯片擦除或块擦除功能才能将代码保护位置 1。而该功能又仅能通过 ICSP 或外部编程器启用。

图 25-6： 禁止表写操作 （**WRTn**）

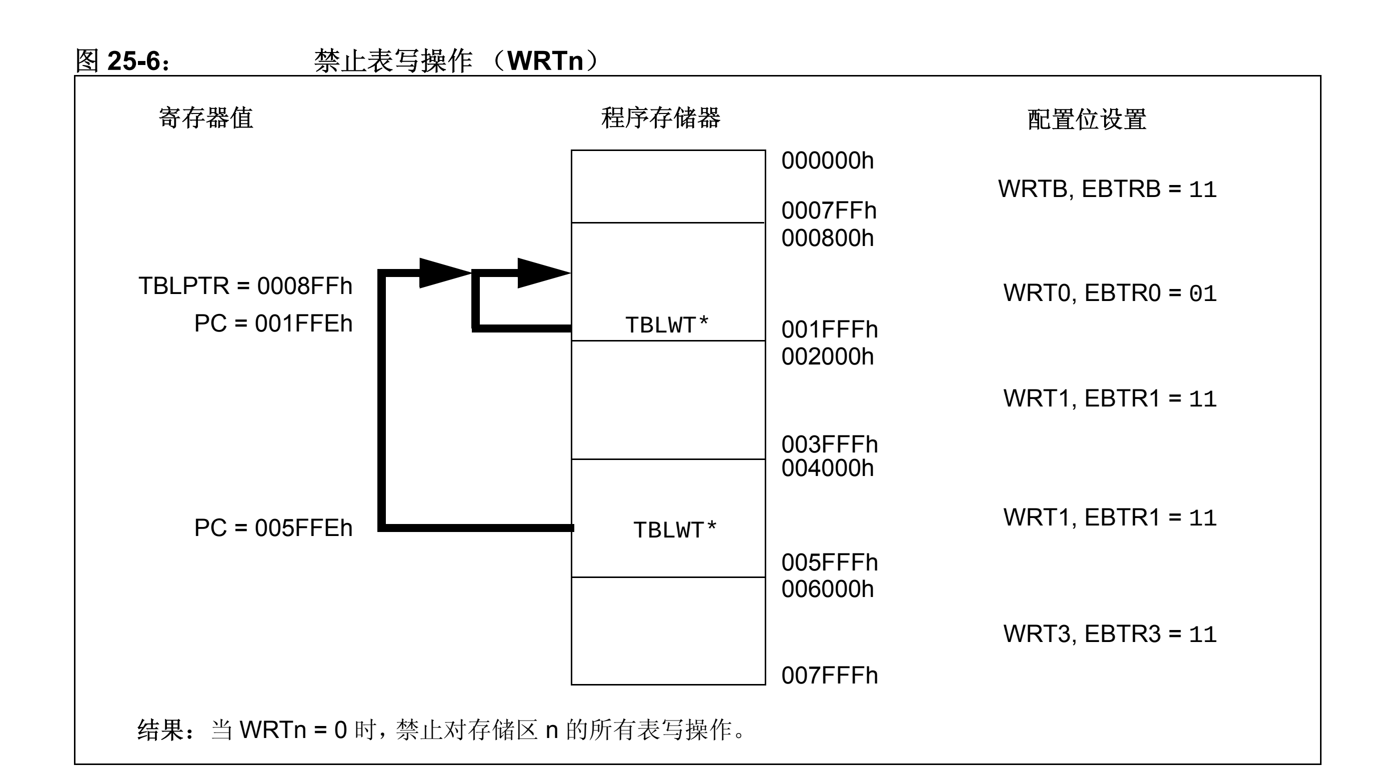


图 25-7: 禁止外部存储区表读操作 (EBTRn)

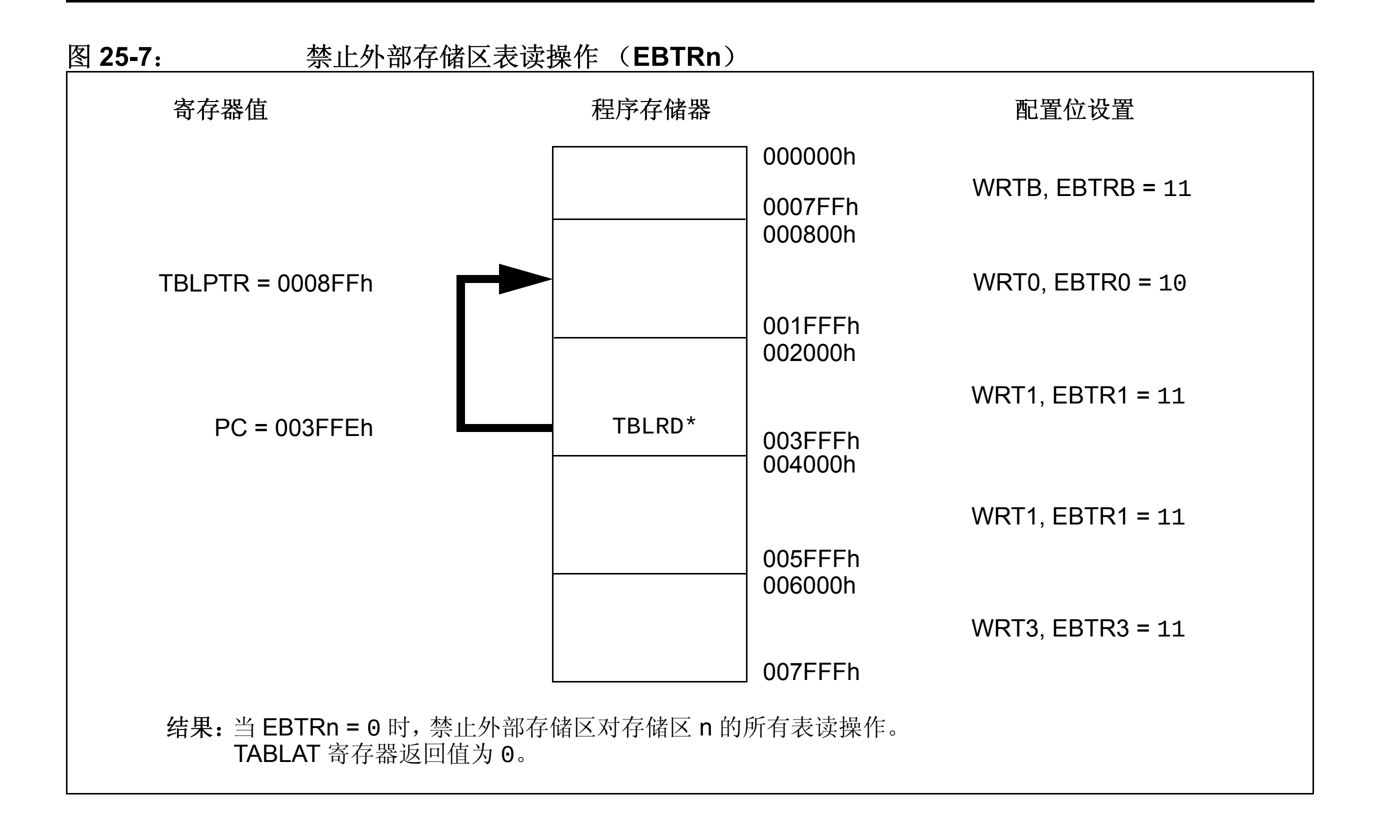
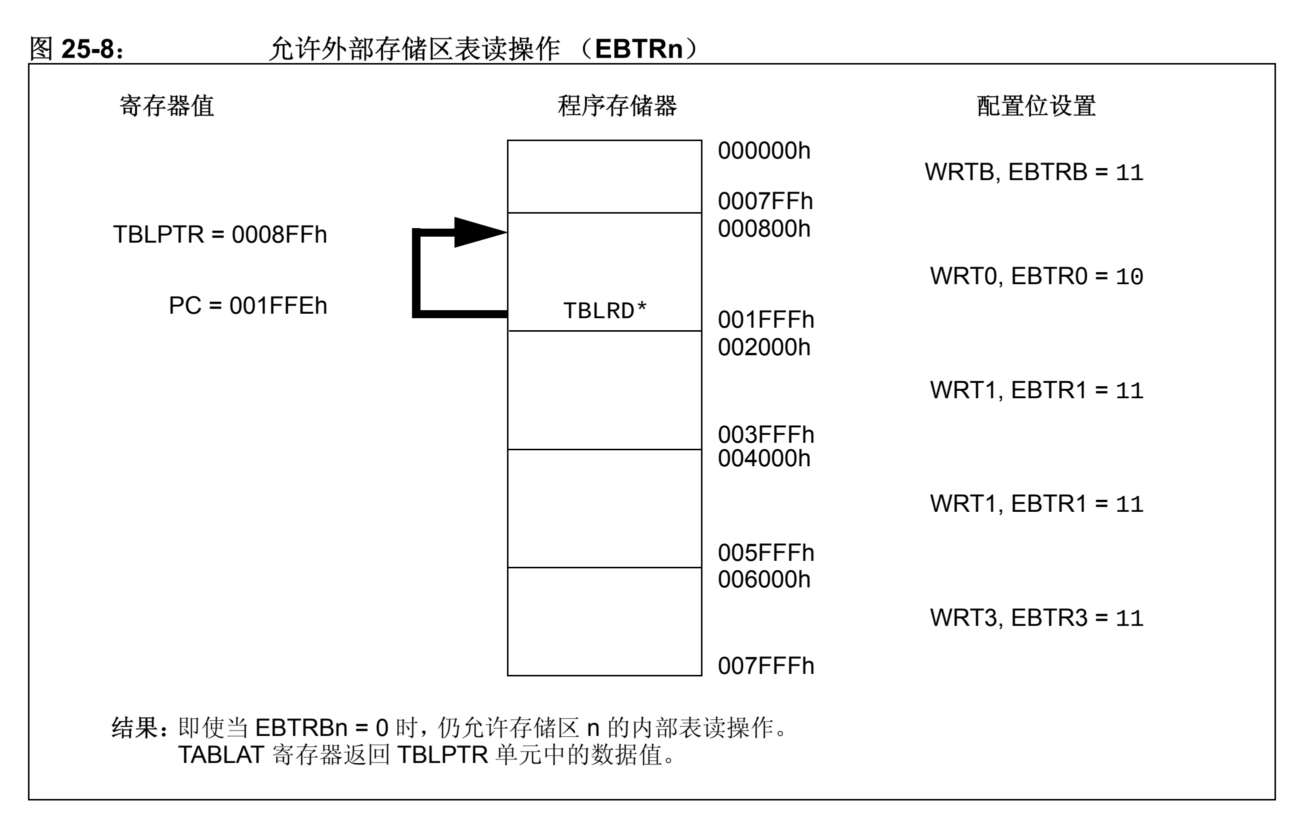


图 25-8: 允许外部存储区表读操作 (EBTRn)



 初稿

25.5.2 数据 EEPROM 代码保护

整个数据 EEPROM 的外部读写保护由 CPD 和 WRTD 这两个位控制。 CPD 禁止外部读写数据 EEPROM。WRTD 禁止内部和外部写入数据 EEPROM。不管保护位如何设置， CPU 仍能读写数据 EEPROM。

25.5.3 配置寄存器保护

配置寄存器可以被写保护。WRTC 位控制对配置寄存器的写保护。在正常执行模式下， WRTC 位是只读的。WRTC 位仅能通过 ICSP 或外部编程器修改。

25.6 ID 单元

有 8 个存储单元 （200000h-200007h）被指定为 ID 单元，供用户存储校验和或其他代码标识。在正常执行期间可通过 TBLRD 和 TBLWT 指令读写这些单元；也可通过编程 / 验证读写这些单元。当器件有代码保护时，仍可读取 ID 单元。

25.7 在线串行编程

PIC18F2455/2550/4455/4550 系列单片机可以在最终应用电路中进行串行编程。只需要 5 根线即可完成这一操作，其中时钟线、数据线各一根，其余 3 根分别是电源线、地线和编程电压线。这允许用户使用未编程器件制造电路板，仅在产品交付前才对单片机进行编程。从而保证可使用最新版本的固件或定制固件。

25.8 在线调试器

将 DEBUG 配置位清 0 可使能在线调试功能。这一功能允许使用 MPLAB® IDE 进行一些简单的调试。当使能了单片机的这项功能时，某些资源就不再是通用的了。表 25-4 显示了后台调试器所需的资源。

表 25-4： 调试器资源

I/O 引脚：	RB6 和 RB7
堆栈：	2 级
程序存储器：	512 字节
数据存储器：	10 字节

要使用单片机的在线调试器功能，用户在设计中就必须实现与 $\overline{\text{MCLR}}$/V_{PP}/RE3、V_{DD}、V_{SS}、RB7 和 RB6 引脚的正确连接，从而与 Microchip 或第三方开发工具公司提供的在线调试器模块交互。

25.9 ICPORT 特殊功能 （仅指定的封装）

在特定的环境下，44引脚TQFP封装的PIC18F4455/4550 器件的NC引脚能提供其他功能。这些功能由器件配置位控制，仅在该封装类型和引脚数下可用。

25.9.1 专用 ICD/ICSP 端口

44 引脚 TQFP 器件能使用 NC 引脚为在线调试（ICD）和在线串行编程 （ICSP）提供备用端口。这些引脚通称为专用 ICSP/ICD 端口，因为它们不与器件的其他功能复用。

在使用时，专用端口激活 3 个 NC 引脚，提供额外的器件复位、数据和时钟端口。这些端口不与标准 I/O 引脚重叠，从而可将节省的 I/O 引脚供用户设计使用。

通过将 ICPRT 配置位置 1，使能专用 ICSP/ICD 端口。该端口与RB6/RB7上的传统ICSP/ICD端口工作方式相同。表 25-5 列出了用于 ICSP 和 ICD 功能的引脚。

表 25-5： 传统和专用 ICD/ICSP™ 端口内等同的引脚

引脚名称		引脚类型	引脚功能
传统端口	专用端口		
$\overline{\text{MCLR}}$/V_{PP}/RE3	NC/$\overline{\text{ICRST}}$/ICV$_{PP}$	P	器件复位和编程使能
RB6/KBI2/PGC	NC/ICCK/ICPGC	I	串行时钟
RB7/KBI3/PGD	NC/ICDT/ICPGD	I/O	串行数据

图注： I = 输入， O = 输出， P = 电源

即使使能了专用端口，在传统端口上的 ICSP 和 ICD 引脚功能仍然可用。当 $\overline{MCLR}$/V_{PP}/RE3 引脚上检测到 V_{IH} 时， ICRST/ICV_{PP} 引脚的状态将被忽略。

> 注 1：只能通过默认的 ICSP 端口编程 ICPORT 配置位。
>
> 2：28/40 引脚的器件必须将 ICPORT 配置位保持清零；否则会出现不可预期的操作。

25.9.2 28 引脚仿真

44 引脚 TQFP 封装的 PIC18F4455/4550 在外部控制下还能更改配置，用于系统调试。这样器件就能仿真 PIC18F2455/2550 系列的 28 引脚器件。

通过一个 NC/ICPORTS 引脚控制 28 引脚配置模式。把该引脚连到 V_{SS}，强制器件仿真 28 引脚器件；与 40/44 引脚器相关的功能和对应的控制寄存器和位将被禁止。这些功能包括 PORTD 和 PORTE， SSP，以及 CCP1 的增强型 PWM 功能。而把 NC/ICPORTS 连到 V_{DD}，则强制器件工作在默认配置下。

只有在后台调试和专用 ICD/ICSP 端口都使能的情况下（DEBUG 配置位清零，ICPRT 配置位置 1），才可使用配置选项。当被禁止时，NC/ICPORTS 是一个无连接引脚。

25.10 单电源 ICSP 编程

LVP 配置位使能单电源 ICSP 编程（正式名称为*低压 ICSP 编程*或 *LVP*）。当使能单电源编程模式时，不再需要在 $\overline{MCLR}$/V_{PP}/RE3 引脚接入高压，此时 RB5/KBI1/PGM 引脚专用于控制进入编程模式而不再用作通用 I/O 引脚。

当使用单电源编程模式进行编程期间，跟正常执行模式一样， V_{DD} 加到 $\overline{MCLR}$/V_{PP}/RE3 引脚。而在进入编程模式时，则需将 V_{DD} 加到 PGM 引脚。

> 注 1：不管 LVP 位状态如何，只要把 V_{IHH} 电压加在 $\overline{MCLR}$ 引脚上，就可以实现高压编程。
>
> 2：在低压 ICSP 编程模式下， RB5 引脚不再用作通用 I/O 引脚，在正常操作过程中应保持低电平。
>
> 3：当使用低电压 ICSP 编程（LVP），并启用 PORTB 的上拉电阻时，必须将 TRISB 寄存器的第 5 位清零以禁止 RB5 的上拉电路，从而保证器件正常工作。
>
> 4：如果禁止了器件的主复位，需验证是否完成下面任一操作，以保证能正常进入ICSP 模式：
>
> a) 禁止低压编程（CONFIG4L<2> = 0）；或
>
> b) 在进入 ICSP 时保证 RB5/KBI1/PGM 为低电平。

如果不使用单电源 ICSP 编程模式，可以将 LVP 位清零。RB5/KBI1/PGM 则可用作数字 I/O 引脚 RB5。仅当使用标准高压编程模式（将 V_{IHH} 电压加在 $\overline{MCLR}$/V_{PP}/RE3 引脚上）时，可以清零或置位 LVP 位。一旦禁止了 LVP，只有标准高压编程模式可用，且必须用该模式来为器件编程。

没有代码保护的存储器可用块擦除或逐行擦除的方法来擦除，然后可在指定的 V_{DD} 电压下对其进行写操作。如果要擦除有代码保护的存储器，则必须使用块擦除。当在低压编程模式下执行块擦除时，必须为器件提供 4.5V 到 5.5V 的 V_{DD}。

26.0 指令集综述

PIC18F2455/2550/4455/4550 器件具有一个含有 75 个 PIC18 内核指令的标准指令集，和一个含有 8 个新指令（优化递归和软件堆栈代码）的扩展指令集。本章后面部分将讨论该扩展指令集。

26.1 标准指令集

标准的 PIC18 指令集与以前的 PICmicro 指令集相比，添加了很多增强功能，并保持了易于从 PICmicro 指令集移植的特点。大部分指令为单字指令（16 位），只有 4 条指令是双字指令。

每条单字指令都是一个 16 位字，由操作码（指明指令类型）和一个或多个操作数（指定指令操作）组成。

整个指令集具有高度的正交性，分为以下4种基本类型：

- **字节**操作类指令
- **位**操作类指令
- **立即数**操作类指令
- **控制**操作类指令

表 26-2 为 PIC18 指令集汇总，它列出了**字节**、**位**以及**立即数**和**控制**操作类指令。表 26-1 给出了对操作码字段的说明。

大部分**字节**操作类指令含有三种操作数：

1. 文件寄存器（由“f”指定）
2. 保存结果的目标寄存器（由“d”指定）
3. 被访问的存储器（由“a”指定）

文件寄存器指示符“f”指定了指令将会使用哪一个文件寄存器。目标寄存器指示符“d”指定了操作结果的存放位置。如果“d”为 0，操作结果存入 WREG 寄存器中；如果“d”为 1，操作结果存入指令指定的文件寄存器中。

所有**位**操作类指令都含有三种操作数：

1. 文件寄存器（由“f”指定）
2. 文件寄存器中的位（由“b”指定）
3. 被访问的存储器（由“a”指定）

位域指示符“b”选择操作所影响的位的编号，而文件寄存器指示符“f”则代表这些位所在的寄存器的编号。

立即数操作指令使用以下操作数：

- 要装入文件寄存器中的立即数（由“k”指定）
- 要装入立即数的 FSR 寄存器（由“f”指定）
- 不需要操作数（由“—”指定）

控制操作类指令使用以下操作数：

- 程序存储器地址（由“n”指定）
- CALL 或 RETURN 指令的模式（由“s”指定）
- 表读和表写指令的模式（由“m”指定）
- 不需要操作数（由“—”指定）

除了 4 条双字指令外，所有的指令都是单字指令。双字指令将所需的信息保存在 32 位中。第二个字的高 4 位都是1。如果第二个字作为一条指令执行，它会执行NOP 指令。

除非条件测试结果为“true”或者指令执行改变了程序计数器的值，否则执行所有的单字指令都只需要一个指令周期。对于上述两种特殊情况，执行指令需要两个指令周期，第二个指令周期执行一条 NOP 指令。

执行双字指令需要两个指令周期。

每个指令周期由 4 个振荡周期组成。因此，对于频率为 4 MHz 的振荡器，其正常的指令执行时间为 1 µs。如果条件测试为“true”或指令执行改变了程序计数器值，则该指令的执行时间为 2 µs。双字转移指令（如果为“true”）的执行需要 3 µs。

图 26-1 给出了指令的几种通用格式。所有示例均使用“nnh”来表示十六进制数。

指令集汇总（见表 26-2）列出了可被 Microchip MPASM™ 汇编器识别的标准指令。

第 **26.1.1 节“标准指令集”**中对每条指令进行了介绍。

表 26-1： 操作码字段说明

字段	说明
a	快速操作 RAM 位 a = 0：快速操作 RAM 内的 RAM 单元 （BSR 寄存器被忽略） a = 1：由 BSR 寄存器指定的 RAM 存储区
bbb	8 位文件寄存器内的位地址 （0 到 7）。
BSR	存储区选择寄存器。用于选择当前的 RAM 存储区。
C、 DC、 Z、 OV 和 N	ALU 状态位：进位标志位、辅助进位标志位、全零标志位、溢出标志位和负标志位
d	目标寄存器选择位 d = 0：结果保存至 WREG 寄存器 d = 1：结果保存至文件寄存器 f
dest	目标寄存器：可以是 WREG 寄存器或指定的寄存器单元。
f	8 位寄存器地址 （00h 到 FFh）或 2 位 FSR 指示符 （0h 到 3h）。
f_s	12 位寄存器地址 （000h 到 FFFh）。这是源地址。
f_d	12 位寄存器地址 （000h 到 FFFh）。这是目标地址。
GIE	全局中断允许位。
k	立即数字段、常数或者标号 （可能是 8 位、 12 位或 20 位的值）。
label	标号名称。
mm * *+ *- +*	表读和表写指令的 TBLPTR 寄存器模式。 只和表读和表写指令一起使用： 不改变寄存器 （如用于表读和表写的 TBLPTR） 后增寄存器 （如用于表读和表写的 TBLPTR） 后减寄存器 （如用于表读和表写的 TBLPTR） 预增寄存器 （如用于表读和表写的 TBLPTR）
n	相对跳转指令的相对地址 （二进制补码），或 Call/Branch 和 Return 指令的直接地址。
PC	程序计数器。
PCL	程序计数器的低字节。
PCH	程序计数器的高字节。
PCLATH	程序计数器的高字节锁存器。
PCLATU	程序计数器的最高字节锁存器。
$\overline{PD}$	掉电位。
PRODH	乘积的高字节。
PRODL	乘积的低字节。
s	快速调用 / 返回模式选择位 s = 0：不对影子寄存器进行更新，也不用影子寄存器的内容更新其他寄存器 s = 1：将某些寄存器的值存入影子寄存器或把影子寄存器的值载入某些寄存器 （快速模式）
TBLPTR	21 位表指针 （指向程序存储单元）。
TABLAT	8 位表锁存器。
$\overline{TO}$	超时溢出位。
TOS	栈顶。
u	未使用或不变。
WDT	看门狗定时器。
WREG	工作寄存器 （累加器）。
x	任意值 （0 或 1）。汇编器将产生 x = 0 的代码。为了与所有的 Microchip 软件工具兼容，建议使用这种格式。
z_s	对寄存器 （源）进行间接寻址的 7 位偏移量。
z_d	对寄存器 （目标）进行间接寻址的 7 位偏移量。
{ }	可选参数。
[text]	表示变址地址。
(text)	text 的内容。
[expr]<n>	表示由指针 expr 指定的寄存器中的位 n。
→	赋值。
< >	寄存器位域。
∈	表示属于某个集合。
斜体文字	用户定义项 （字体为 Courier）。

图 26-1： 指令的通用格式

	指令示例
面向字节的文件寄存器操作	

15 10 9 8 7 0

操作码	d	a	f（寄存器地址）

ADDWF MYREG, W, B

d = 0 表示结果存入 WREG 寄存器
d = 1 表示结果存入文件寄存器（f）
a = 0 强制使用快速操作存储区
a = 1 使用 BSR 选择存储区
f = 8 位文件寄存器地址

字节到字节的传送操作（双字）

15 12 11 0

操作码	f（源寄存器地址）

15 12 11 0

1111	f（目标寄存器地址）

MOVFF MYREG1, MYREG2

f = 12 位文件寄存器地址

面向位的文件寄存器操作

15 12 11 9 8 7 0

操作码	b（位地址）	a	f（寄存器地址）

BSF MYREG, bit, B

b = 占 3 位，表示文件寄存器（f）中位的位置
a = 0 强制使用快速操作存储区
a = 1 使用 BSR 选择存储区
f = 8 位文件寄存器地址

立即数操作

15 8 7 0

操作码	k （立即数）

MOVLW 7Fh

k = 8 位立即数的值

控制操作

CALL、GOTO 和跳转类操作

15 8 7 0

操作码	n<7:0>（立即数）

15 12 11 0

1111	n<19:8>（立即数）

GOTO 标号

n = 20 位立即数的值

15 8 7 0

操作码	S	n<7:0>（立即数）

15 12 11 0

1111	n<19:8>（立即数）

CALL MYFUNC

S = 快速位

15 11 10 0

操作码	n<10:0>（立即数）

BRA MYFUNC

15 8 7 0

操作码	n<7:0>（立即数）

BC MYFUNC

表 26-2： PIC18FXXXX 指令集

助记符	操作数	说明	周期数	16 位指令字 MSb			LSb	受影响的状态位	注
面向字节的操作类指令									
ADDWF	f, d, a	WREG 与 f 相加	1	0010	01da	ffff	ffff	C、DC、Z、OV 和 N	1, 2
ADDWFC	f, d, a	WREG 与 f 带进位相加	1	0010	00da	ffff	ffff	C、DC、Z、OV 和 N	1, 2
ANDWF	f, d, a	WREG 与 f 作逻辑与运算	1	0001	01da	ffff	ffff	Z 和 N	1, 2
CLRF	f, a	将 f 清零	1	0110	101a	ffff	ffff	Z	2
COMF	f, d, a	将 f 取反	1	0001	11da	ffff	ffff	Z 和 N	1, 2
CPFSEQ	f, a	将 f 与 WREG 作比较，相等则跳过	1 （2 或 3）	0110	001a	ffff	ffff	无	4
CPFSGT	f, a	将 f 与 WREG 作比较，大于则跳过	1 （2 或 3）	0110	010a	ffff	ffff	无	4
CPFSLT	f, a	将 f 与 WREG 作比较，小于则跳过	1 （2 或 3）	0110	000a	ffff	ffff	无	1, 2
DECF	f, d, a	f 减 1	1	0000	01da	ffff	ffff	C、DC、Z、OV 和 N	1, 2, 3, 4
DECFSZ	f, d, a	f 减 1，为 0 则跳过	1 （2 或 3）	0010	11da	ffff	ffff	无	1, 2, 3, 4
DCFSNZ	f, d, a	f 减 1，非 0 则跳过	1 （2 或 3）	0100	11da	ffff	ffff	无	1, 2
INCF	f, d, a	f 加 1	1	0010	10da	ffff	ffff	C、DC、Z、OV 和 N	1, 2, 3, 4
INCFSZ	f, d, a	f 加 1，为 0 则跳过	1 （2 或 3）	0011	11da	ffff	ffff	无	4
INFSNZ	f, d, a	f 加 1，非 0 则跳过	1 （2 或 3）	0100	10da	ffff	ffff	无	1, 2
IORWF	f, d, a	WREG 与 f 作逻辑或运算	1	0001	00da	ffff	ffff	Z 和 N	1, 2
MOVF	f, d, a	传送 f	1	0101	00da	ffff	ffff	Z 和 N	1
MOVFF	f_s, f_d	将 f_s （源）送入第一个字	2	1100	ffff	ffff	ffff	无	
		将 f_d （目标）送入第二个字		1111	ffff	ffff	ffff		
MOVWF	f, a	将 WREG 传送给 f	1	0110	111a	ffff	ffff	无	
MULWF	f, a	WREG 乘以 f	1	0000	001a	ffff	ffff	无	1, 2
NEGF	f, a	对 f 取补	1	0110	110a	ffff	ffff	C、DC、Z、OV 和 N	
RLCF	f, d, a	f 带进位循环左移	1	0011	01da	ffff	ffff	C、Z 和 N	1, 2
RLNCF	f, d, a	f 循环左移 （不带进位）	1	0100	01da	ffff	ffff	Z 和 N	
RRCF	f, d, a	f 带进位循环右移	1	0011	00da	ffff	ffff	C、Z 和 N	
RRNCF	f, d, a	f 循环右移 （不带进位）	1	0100	00da	ffff	ffff	Z 和 N	
SETF	f, a	将 f 的内容置为全 1	1	0110	100a	ffff	ffff	无	1, 2
SUBFWB	f, d, a	WREG 减去 f （带借位）	1	0101	01da	ffff	ffff	C、DC、Z、OV 和 N	
SUBWF	f, d, a	f 减去 WREG	1	0101	11da	ffff	ffff	C、DC、Z、OV 和 N	1, 2
SUBWFB	f, d, a	f 减去 WREG （带借位）	1	0101	10da	ffff	ffff	C、DC、Z、OV 和 N	
SWAPF	f, d, a	将 f 中的两个半字节进行交换	1	0011	10da	ffff	ffff	无	4
TSTFSZ	f, a	f 为 0 则跳过	1 （2 或 3）	0110	011a	ffff	ffff	无	1, 2
XORWF	f, d, a	WREG 与 f 作逻辑异或运算	1	0001	10da	ffff	ffff	Z 和 N	

注 **1：** Port 寄存器的值随端口状态的变化而不断修改 （例如， `MOVF PORTB, 1, 0`），修改时使用的值是引脚上的当前值。例如，如果将一引脚配置为输入，虽然其数据锁存器中的值为 `1`，但此时外部器件将该引脚驱动为低电平，则被写回数据总线的数据值将是 `0`。

2： 当对 TMR0 寄存器 （以及其他适用的寄存器）执行该指令时 （并且 d = `1`），如果已为其分配了预分频器，则将对该预分频器清零。

3： 如果程序计数器 （PC）被修改或者条件判断为 “true”，则该指令需要两个周期。第二个周期执行一条 `NOP` 指令。

4： 某些指令是双字指令。除非指令的第一个字获取这 16 位中包含的信息，否则第二个字将作为 `NOP` 指令执行。这将确保所有程序存储单元内存储的都是合法的指令。

表 **26-2：** **PIC18FXXXX 指令集** （续）

助记符	操作数	说明	周期数	16 位指令字				受影响的状态位	注
				MSb			**LSb**		
面向位的操作类指令									
BCF	f, b, a	将 f 寄存器中的某位清零	1	1001	bbba	ffff	ffff	无	1, 2
BSF	f, b, a	将 f 寄存器中的某位置 1	1	1000	bbba	ffff	ffff	无	1, 2
BTFSC	f, b, a	检测 f 中的某位，为 0 则跳过	1 （2 或 3）	1011	bbba	ffff	ffff	无	3, 4
BTFSS	f, b, a	检测 f 中的某位，为 1 则跳过	1 （2 或 3）	1010	bbba	ffff	ffff	无	3, 4
BTG	f, d, a	将 f 中的某位取反	1	0111	bbba	ffff	ffff	无	1, 2
控制操作类指令									
BC	n	进位则跳转	1 （2）	1110	0010	nnnn	nnnn	无	
BN	n	为负则跳转	1 （2）	1110	0110	nnnn	nnnn	无	
BNC	n	无进位则跳转	1 （2）	1110	0011	nnnn	nnnn	无	
BNN	n	不为负则跳转	1 （2）	1110	0111	nnnn	nnnn	无	
BNOV	n	不溢出则跳转	1 （2）	1110	0101	nnnn	nnnn	无	
BNZ	n	不为零则跳转	1 （2）	1110	0001	nnnn	nnnn	无	
BOV	n	溢出则跳转	1 （2）	1110	0100	nnnn	nnnn	无	
BRA	n	无条件跳转	2	1101	0nnn	nnnn	nnnn	无	
BZ	n	为零则跳转	1 （2）	1110	0000	nnnn	nnnn	无	
CALL	n, s	调用子程序 （第一个字）	2	1110	110s	kkkk	kkkk	无	
		（第二个字）		1111	kkkk	kkkk	kkkk		
CLRWDT	—	将看门狗定时器清零	1	0000	0000	0000	0100	$\overline{TO}$ 和 $\overline{PD}$	
DAW	—	对 WREG 进行十进制调整	1	0000	0000	0000	0111	C	
GOTO	n	跳转到地址 （第一个字）	2	1110	1111	kkkk	kkkk	无	
		（第二个字）		1111	kkkk	kkkk	kkkk		
NOP	—	空操作	1	0000	0000	0000	0000	无	
NOP	—	空操作	1	1111	xxxx	xxxx	xxxx	无	4
POP	—	弹出返回堆栈栈顶 （TOS）的内容	1	0000	0000	0000	0110	无	
PUSH	—	压入返回堆栈栈顶 （TOS）	1	0000	0000	0000	0101	无	
RCALL	n	相对调用	2	1101	1nnn	nnnn	nnnn	无	
RESET		用软件使器件复位	1	0000	0000	1111	1111	全部	
RETFIE	s	使能中断返回	2	0000	0000	0001	000s	GIE/GIEH 和 PEIE/GIEL	
RETLW	k	返回时将立即数送入 WREG	2	0000	1100	kkkk	kkkk	无	
RETURN	s	从子程序返回	2	0000	0000	0001	001s	无	
SLEEP	—	进入待机模式	1	0000	0000	0000	0011	$\overline{TO}$ 和 $\overline{PD}$	

注 **1：** Port 寄存器的值随端口状态的变化而不断修改（例如， `MOVF PORTB, 1, 0`），修改时使用的值是引脚上的当前值。例如，如果将一引脚配置为输入，虽然其数据锁存器中的值为 1，但此时外部器件将该引脚驱动为低电平，则被写回数据总线的数据值将是 0。

2： 当对 TMR0 寄存器 （以及其他适用的寄存器）执行该指令时 （并且 d = 1），如果已为其分配了预分频器，则将对该预分频器清零。

3： 如果程序计数器 （PC）被修改或者条件判断为 “true”，则该指令需要两个周期。第二个周期执行一条 NOP 指令。

4： 某些指令是双字指令。除非指令的第一个字获取这 16 位中包含的信息，否则第二个字将作为 NOP 指令执行。这将确保所有程序存储单元内存储的都是合法的指令。

表 26-2： PIC18FXXXX 指令集 （续）

助记符	操作数	说明	周期数	16 位指令字 MSb			LSb	受影响的状态位	注
立即数操作类指令									
ADDLW	k	WREG 与立即数相加	1	0000	1111	kkkk	kkkk	C、DC、Z、OV 和 N	
ANDLW	k	立即数与 WREG 作逻辑与运算	1	0000	1011	kkkk	kkkk	Z 和 N	
IORLW	k	立即数与 WREG 作逻辑或运算	1	0000	1001	kkkk	kkkk	Z 和 N	
LFSR	f, k	将 12 位立即数第二个字传送 到 FSR(f) 第一个字	2	1110 1111	1110 0000	00ff kkkk	kkkk kkkk	无	
MOVLB	k	将立即数送入 BSR<3:0>	1	0000	0001	0000	kkkk	无	
MOVLW	k	将立即数送入 WREG	1	0000	1110	kkkk	kkkk	无	
MULLW	k	立即数与 WREG 相乘	1	0000	1101	kkkk	kkkk	无	
RETLW	k	返回时将立即数送入 WREG	2	0000	1100	kkkk	kkkk	无	
SUBLW	k	立即数减去 WREG	1	0000	1000	kkkk	kkkk	C、DC、Z、OV 和 N	
XORLW	k	立即数与 WREG 作逻辑异或运算	1	0000	1010	kkkk	kkkk	Z 和 N	
数据存储器 ↔ 程序存储器操作类指令									
TBLRD*		表读	2	0000	0000	0000	1000	无	
TBLRD*+		后增表读		0000	0000	0000	1001	无	
TBLRD*-		后减表读		0000	0000	0000	1010	无	
TBLRD+*		预增表读		0000	0000	0000	1011	无	
TBLWT*		表写	2	0000	0000	0000	1100	无	
TBLWT*+		后增表写		0000	0000	0000	1101	无	
TBLWT*-		后减表写		0000	0000	0000	1110	无	
TBLWT+*		预增表写		0000	0000	0000	1111	无	

注 1： Port 寄存器的值随端口状态的变化而不断修改 （例如， MOVF PORTB, 1, 0），修改时使用的值是引脚上的当前值。例如，如果将一引脚配置为输入，虽然其数据锁存器中的值为 1，但此时外部器件将该引脚驱动为低电平，则被写回数据总线的数据值将是 0。

2： 当对 TMR0 寄存器 （以及其他适用的寄存器）执行该指令时 （并且 d = 1），如果已为其分配了预分频器，则将对该预分频器清零。

3： 如果程序计数器 （PC）被修改或者条件判断为 “true”，则该指令需要两个周期。第二个周期执行一条 NOP 指令。

4： 某些指令是双字指令。除非指令的第一个字获取这 16 位中包含的信息，否则第二个字将作为 NOP 指令执行。这将确保所有程序存储单元内存储的都是合法的指令。

26.1.1 标准指令集

ADDLW **W 与立即数相加**

语法： ADDLW k

操作数： 0 ≤ k ≤ 255

操作： (W) + k → W

受影响的状态位： N、 OV、 C、 DC 和 Z

机器码：

0000	1111	kkkk	kkkk

说明： 将 W 的内容与 8 位立即数 k 相加，结果保存在 W 寄存器中。

指令字数： 1

指令周期数： 1

Q 周期操作：

Q1	Q2	Q3	Q4
译码	读立即数 k	处理数据	写入 W

示例： `ADDLW 15h`

执行指令前
W = 10h
执行指令后
W = 25h

ADDWF **W 与 f 寄存器相加**

语法： ADDWF f {,d {,a}}

操作数： 0 ≤ f ≤ 255
d ∈ [0,1]
a ∈ [0,1]

操作： (W) + (f) → dest

受影响的状态位： N、 OV、 C、 DC 和 Z

机器码：

0010	01da	ffff	ffff

说明： 将 W 的内容与 f 寄存器的内容相加。如果 d 为 0，结果存储在 W 中。如果 d 为 1，结果存回寄存器 f （默认）。
如果 a 为 0，选择快速操作存储区。如果 a 为 1，使用 BSR 选择 GPR 存储区（默认）。
如果 a 为 0 且使能了扩展的指令集，只要 f ≤ 95 （5Fh），指令就将以立即数变址寻址模式进行操作。详情请参见**第 26.2.3 节 “立即数变址寻址模式中面向字节和位的指令”**。

指令字数： 1

指令周期数： 1

Q 周期操作：

Q1	Q2	Q3	Q4
译码	读 寄存器 f	处理数据	写入 目标寄存器

示例： `ADDWF REG, 0, 0`

执行指令前
W = 17h
REG = 0C2h
执行指令后
W = 0D9h
REG = 0C2h

注：	所有的 PIC18 指令都可能在其指令助记符之前使用标号参数，用于符号寻址。如果使用了标号，那么指令语法将变为：{ 标号 } 指令参数。

ADDWFC　　W 与 f 带进位相加

语法：	ADDWFC　f {,d {,a}}
操作数：	0 ≤ f ≤ 255 d ∈ [0,1] a ∈ [0,1]
操作：	(W) + (f) + (C) → dest
受影响的状态位：	N、 OV、 C、 DC 和 Z
机器码：	0010 / 00da / ffff / ffff
说明：	将 W 的内容、进位标志位与寄存器 f 的内容相加。如果 d 为 0，结果存储在 W 中。如果 d 为 1，结果存储在寄存器 f 中。 如果 a 为 0，选择快速操作存储区。如果 a 为 1，使用 BSR 选择 GPR 存储区（默认）。 如果 a 为 0 且使能了扩展的指令集，只要 f ≤ 95 （5Fh），指令就将以立即数变址寻址模式进行操作。详情请参见**第 26.2.3 节 "立即数变址寻址模式中面向字节和位的指令"**。
指令字数：	1
指令周期数：	1

Q 周期操作：

Q1	Q2	Q3	Q4
译码	读 寄存器 f	处理数据	写入 目标寄存器

示例：　ADDWFC　REG, 0, 1

执行指令前
　进位标志位 = 1
　REG = 02h
　W = 4Dh

执行指令后
　进位标志位 = 0
　REG = 02h
　W = 50h

ANDLW　　立即数与 W 寄存器作逻辑与运算

语法：	ANDLW　k
操作数：	0 ≤ k ≤ 255
操作：	(W) .AND. k → W
受影响的状态位：	N 和 Z
机器码：	0000 / 1011 / kkkk / kkkk
说明：	将 W 的内容与 8 位立即数 k 作逻辑与运算。结果保存在 W 寄存器中。
指令字数：	1
指令周期数：	1

Q 周期操作：

Q1	Q2	Q3	Q4
译码	读立即数 k	处理数据	写入 W

示例：　ANDLW　05Fh

执行指令前
　W = A3h

执行指令后
　W = 03h

ANDWF 将 W 与 f 作逻辑与运算

语法： ANDWF f {,d {,a}}

操作数： 0 ≤ f ≤ 255
d ∈ [0,1]
a ∈ [0,1]

操作： (W) .AND.(f) → dest

受影响的状态位： N 和 Z

机器码：

0001	01da	ffff	ffff

说明： 将 W 的内容与寄存器 f 的内容作逻辑与运算。如果 d 为 0，结果存储在 W 中。如果 d 为 1，结果存回寄存器 f（默认）。
如果 a 为 0，选择快速操作存储区。如果 a 为 1，使用 BSR 选择 GPR 存储区（默认）。
如果 a 为 0 且使能了扩展的指令集，只要 f ≤ 95 （5Fh），指令就将以立即数变址寻址模式进行操作。详情请参见**第 26.2.3 节“立即数变址寻址模式中面向字节和位的指令”**。

指令字数： 1

指令周期数： 1

Q 周期操作：

Q1	Q2	Q3	Q4
译码	读寄存器 f	处理数据	写入目标寄存器

示例： ANDWF REG, 0, 0

执行指令前
W = 17h
REG = C2h
执行指令后
W = 02h
REG = C2h

BC 进位则跳转

语法： BC n

操作数： -128 ≤ n ≤ 127

操作： 如果进位标志位为 1
(PC) + 2 + 2n → PC

受影响的状态位： 无

机器码：

1110	0010	nnnn	nnnn

说明： 如果进位标志位为 1，那么程序将跳转。二进制补码 “2n”与 PC 相加。因为 PC 将递增以便取出下一条指令，所以新地址将为 PC + 2 + 2n。这种情况下，该指令是一条双周期指令。

指令字数： 1

指令周期数： 1 （2）

Q 周期操作：
如果跳转：

Q1	Q2	Q3	Q4
译码	读立即数 n	处理数据	写入 PC
空操作	空操作	空操作	空操作

如果不跳转：

Q1	Q2	Q3	Q4
译码	读立即数 n	处理数据	空操作

示例： HERE BC 5

执行指令前
PC = 地址 （HERE）
执行指令后
如果进位标志位 = 1；
PC = 地址 （HERE + 12）
如果进位标志位 = 0；
PC = 地址 （HERE + 2）

BCF　将 f 寄存器中的某位清零

语法：　BCF　f, b {,a}

操作数：　0 ≤ f ≤ 255
0 ≤ b ≤ 7
a ∈ [0,1]

操作：　0 → f<b>

受影响的状态位：　无

机器码：

1001	bbba	ffff	ffff

说明：　将寄存器 f 中的位 b 清零。
如果 a 为 0，选择快速操作存储区。如果 a 为 1，使用 BSR 选择 GPR 存储区（默认）。
如果 a 为 0 且使能了扩展的指令集，只要 f ≤ 95（5Fh），指令就将以立即数变址寻址模式进行操作。详情请参见**第 26.2.3 节“立即数变址寻址模式中面向字节和位的指令”**。

指令字数：　1

指令周期数：　1

Q 周期操作：

Q1	Q2	Q3	Q4
译码	读寄存器 f	处理数据	写寄存器 f

示例：　`BCF　FLAG_REG,　7, 0`

执行指令前
FLAG_REG =　C7h

执行指令后
FLAG_REG =　47h

BN　为负则跳转

语法：　BN　n

操作数：　-128 ≤ n ≤ 127

操作：　如果负标志位为 1
(PC) + 2 + 2n → PC

受影响的状态位：　无

机器码：

1110	0110	nnnn	nnnn

说明：　如果负标志位为 1，那么 程序将跳转。二进制补码 “2n”与 PC 相加。因为 PC 将递增以便取出下一条指令，所以新地址将为 PC + 2 + 2n。这种情况下，该指令是一条双周期指令。

指令字数：　1

指令周期数：　1（2）

Q 周期操作：

如果跳转：

Q1	Q2	Q3	Q4
译码	读立即数 n	处理数据	写入 PC
空操作	空操作	空操作	空操作

如果不跳转：

Q1	Q2	Q3	Q4
译码	读立即数 n	处理数据	空操作

示例：　`HERE　BN　Jump`

执行指令前
PC　=　地址（HERE）

执行指令后
如果负标志位　=　1；
PC　=　地址（Jump）
如果负标志位　=　0；
PC　=　地址（HERE + 2）

BNC 无进位则跳转

语法： BNC n

操作数： -128 ≤ n ≤ 127

操作： 如果进位标志位为 0
(PC) + 2 + 2n → PC

受影响的状态位： 无

机器码：

1110	0011	nnnn	nnnn

说明： 如果进位标志位为 0，那么程序将跳转。二进制补码 “2n” 与 PC 相加。因为 PC 将递增以便取出下一条指令，所以新地址将为 PC + 2 + 2n。这种情况下，该指令是一条双周期指令。

指令字数： 1

指令周期数： 1（2）

Q 周期操作：
如果跳转：

Q1	Q2	Q3	Q4
译码	读立即数 n	处理数据	写入 PC
空操作	空操作	空操作	空操作

如果不跳转：

Q1	Q2	Q3	Q4
译码	读立即数 n	处理数据	空操作

示例： HERE BNC Jump

执行指令前
PC = 地址 （HERE）
执行指令后
如果进位标志位= 0；
PC = 地址 （Jump）
如果进位标志位= 1；
PC = 地址 （HERE + 2）

BNN 不为负则跳转

语法： BNN n

操作数： -128 ≤ n ≤ 127

操作： 如果负标志位为 0
(PC) + 2 + 2n → PC

受影响的状态位： 无

机器码：

1110	0111	nnnn	nnnn

说明： 如果负标志位为 0，那么程序将跳转。二进制补码 “2n” 与 PC 相加。因为 PC 要先递增以便取出下一条指令，所以新地址将为 PC + 2 + 2n。这种情况下，该指令是一条双周期指令。

指令字数： 1

指令周期数： 1（2）

Q 周期操作：
如果跳转：

Q1	Q2	Q3	Q4
译码	读立即数 n	处理数据	写入 PC
空操作	空操作	空操作	空操作

如果不跳转：

Q1	Q2	Q3	Q4
译码	读立即数 n	处理数据	空操作

示例： HERE BNN Jump

执行指令前
PC = 地址 （HERE）
执行指令后
如果负标志位 = 0；
PC = 地址 （Jump）
如果负标志位 = 1；
PC = 地址 （HERE + 2）

BNOV 不溢出则跳转

语法： BNOV n

操作数： -128 ≤ n ≤ 127

操作： 如果溢出标志位为 0
(PC) + 2 + 2n → PC

受影响的状态位： 无

机器码：

1110	0101	nnnn	nnnn

说明： 如果溢出标志位为 0，那么程序将跳转。二进制补码 “2n” 与 PC 相加。因为 PC 要先递增以便取出下一条指令，所以新地址将为 PC + 2 + 2n。这种情况下，该指令是一条双周期指令。

指令字数： 1

指令周期数： 1（2）

Q 周期操作：
如果跳转：

Q1	Q2	Q3	Q4
译码	读立即数 n	处理数据	写入 PC
空操作	空操作	空操作	空操作

如果不跳转：

Q1	Q2	Q3	Q4
译码	读立即数 n	处理数据	空操作

示例： HERE BNOV Jump

执行指令前
PC = 地址 （HERE）
执行指令后
如果溢出标志位 = 0；
PC = 地址 （Jump）
如果溢出标志位 = 1；
PC = 地址 （HERE + 2）

BNZ 不为零则跳转

语法： BNZ n

操作数： -128 ≤ n ≤ 127

操作： 如果全零标志位为 0
(PC) + 2 + 2n → PC

受影响的状态位： 无

机器码：

1110	0001	nnnn	nnnn

说明： 如果全零标志位为 0，那么程序将跳转。二进制补码 “2n” 与 PC 相加。因为 PC 要先递增以便取出下一条指令，所以新地址将为 PC + 2 + 2n。这种情况下，该指令是一条双周期指令。

指令字数： 1

指令周期数： 1（2）

Q 周期操作：
如果跳转：

Q1	Q2	Q3	Q4
译码	读立即数 n	处理数据	写入 PC
空操作	空操作	空操作	空操作

如果不跳转：

Q1	Q2	Q3	Q4
译码	读立即数 n	处理数据	空操作

示例： HERE BNZ Jump

执行指令前
PC = 地址 （HERE）
执行指令后
如果全零标志位= 0；
PC = 地址 （Jump）
如果全零标志位= 1；
PC = 地址 （HERE + 2）

BRA **无条件跳转**

语法： BRA n

操作数： $-1024 \leqslant n \leqslant 1023$

操作： $(PC) + 2 + 2n \rightarrow PC$

受影响的状态位： 无

机器码：

1101	0nnn	nnnn	nnnn

说明： 将二进制补码 “2n” 与 PC 相加。因为 PC 要先递增才能取下一条指令，所以新地址将为 PC + 2 + 2n。该指令为一条双周期指令。

指令字数： 1

指令周期数： 2

Q 周期操作：

Q1	Q2	Q3	Q4
译码	读立即数 n	处理数据	写入 PC
空操作	空操作	空操作	空操作

示例：

```
HERE    BRA  Jump
```

执行指令前
PC = 地址 （HERE）
执行指令后
PC = 地址 （Jump）

BSF **将 f 寄存器中的某位置 1**

语法： BSF f, b {,a}

操作数： $0 \leqslant f \leqslant 255$
$0 \leqslant b \leqslant 7$
$a \in [0,1]$

操作： $1 \rightarrow f<b>$

受影响的状态位： 无

机器码：

1000	bbba	ffff	ffff

说明： 将寄存器 f 中的位 b 置 1。
如果 a 为 0，选择快速操作存储区。如果 a 为 1，使用 BSR 选择 GPR 存储区（默认）。
如果 a 为 0 且使能了扩展的指令集，只要 f ≤ 95 （5Fh），指令就将以立即数变址寻址模式进行操作。详情请参见**第 26.2.3 节 “立即数变址寻址模式中面向字节和位的指令”**。

指令字数： 1

指令周期数： 1

Q 周期操作：

Q1	Q2	Q3	Q4
译码	读寄存器 f	处理数据	写寄存器 f

示例：

```
BSF     FLAG_REG,  7,1
```

执行指令前
FLAG_REG = 0Ah
执行指令后
FLAG_REG = 8Ah

BTFSC 测试寄存器中的位，为 0 则跳过

语法： BTFSC f, b {,a}

操作数： 0 ≤ f ≤ 255
0 ≤ b ≤ 7
a ∈ [0,1]

操作： 如果 （f<b>） = 0 则跳过

受影响的状态位： 无

机器码：

1011	bbba	ffff	ffff

说明： 如果寄存器 f 中的位 b 为 0，则跳过下一条指令。即在位 b 为 0 时，丢弃在当前指令执行期间取到的下一条指令，转而执行一条 NOP 指令，使该指令变成双周期指令。
如果 a 为 0，选择快速操作存储区。如果 a 为 1，使用 BSR 选择 GPR 存储区 （默认）。
如果 a 为 0 且使能了扩展的指令集，只要 f ≤ 95 (5Fh)，指令就将以立即数变址寻址模式进行操作。详情请参见**第 26.2.3 节 "立即数变址寻址模式中面向字节和位的指令"**。

指令字数： 1

指令周期数： 1 （2）
注： 如果跳过的指令后面跟有双字指令，则执行该指令需要 3 个周期。

Q 周期操作：

Q1	Q2	Q3	Q4
译码	读寄存器 f	处理数据	空操作

如果跳过：

Q1	Q2	Q3	Q4
空操作	空操作	空操作	空操作

如果跳过的指令后面跟有双字指令：

Q1	Q2	Q3	Q4
空操作	空操作	空操作	空操作
空操作	空操作	空操作	空操作

示例：

```
HERE    BTFSC   FLAG, 1, 0
FALSE   :
TRUE    :
```

执行指令前
PC = 地址 （HERE）

执行指令后
如果 FLAG<1> = 0；
PC = 地址 （TRUE）
如果 FLAG<1> = 1；
PC = 地址 （FALSE）

BTFSS 测试寄存器中的位，为 1 则跳过

语法： BTFSS f, b {,a}

操作数： 0 ≤ f ≤ 255
0 ≤ b < 7
a ∈ [0,1]

操作： 如果 （f<b>） = 1 则跳过

受影响的状态位： 无

机器码：

1010	bbba	ffff	ffff

说明： 如果寄存器 f 中的位 b 为 1，则跳过下一条指令。即在位 b 为 1 时，丢弃在当前指令执行期间取到的下一条指令，转而执行一条 NOP 指令，使该指令变成双周期指令。
如果 a 为 0，选择快速操作存储区。如果 a 为 1，使用 BSR 选择 GPR 存储区 （默认）。
如果 a 为 0 且使能了扩展的指令集，只要 f ≤ 95 (5Fh)，指令就将以立即数变址寻址模式进行操作。详情请参见**第 26.2.3 节 "立即数变址寻址模式中面向字节和位的指令"**。

指令字数： 1

指令周期数： 1 （2）
注： 如果跳过的指令后面跟有双字指令，则执行该指令需要 3 个周期。

Q 周期操作：

Q1	Q2	Q3	Q4
译码	读寄存器 f	处理数据	空操作

如果跳过：

Q1	Q2	Q3	Q4
空操作	空操作	空操作	空操作

如果跳过的指令后面跟有双字指令：

Q1	Q2	Q3	Q4
空操作	空操作	空操作	空操作
空操作	空操作	空操作	空操作

示例：

```
HERE    BTFSS   FLAG, 1, 0
FALSE   :
TRUE    :
```

执行指令前
PC = 地址 （HERE）

执行指令后
如果 FLAG<1> = 0；
PC = 地址 （FALSE）
如果 FLAG<1> = 1；
PC = 地址 （TRUE）

BTG 将 f 中的某位取反

语法：	BTG f, b {,a}
操作数：	0 ≤ f ≤ 255 0 ≤ b < 7 a ∈ [0,1]
操作：	$\overline{(f<b>)}$ → f<b>
受影响的状态位：	无
机器码：	0111 bbba ffff ffff
说明：	将数据存储单元 f 中的位 b 取反。 如果 a 为 0，选择快速操作存储区。如果 a 为 1，使用 BSR 选择 GPR 存储区（默认）。 如果 a 为 0 且使能了扩展的指令集，只要 f ≤ 95（5Fh），指令就将以立即数变址寻址模式进行操作。详情请参见**第 26.2.3 节"立即数变址寻址模式中面向字节和位的指令"**。
指令字数：	1
指令周期数：	1

Q 周期操作：

Q1	Q2	Q3	Q4
译码	读寄存器 f	处理数据	写寄存器 f

示例： `BTG PORTC, 4, 0`

执行指令前：
PORTC = 0111 0101 [75h]
执行指令后：
PORTC = 0110 0101 [65h]

BOV 溢出则跳转

语法：	BOV n
操作数：	-128 ≤ n ≤ 127
操作：	如果溢出标志位为 1 (PC) + 2 + 2n → PC
受影响的状态位：	无
机器码：	1110 0100 nnnn nnnn
说明：	如果溢出标志位为 1，那么程序将跳转。二进制补码 "2n" 与 PC 相加。因为 PC 要先递增以便取出下一条指令，所以新地址将为 PC + 2 + 2n。这种情况下，该指令是一条双周期指令。
指令字数：	1
指令周期数：	1（2）

Q 周期操作：
如果跳转：

Q1	Q2	Q3	Q4
译码	读立即数 n	处理数据	写入 PC
空操作	空操作	空操作	空操作

如果不跳转：

Q1	Q2	Q3	Q4
译码	读立即数 n	处理数据	空操作

示例： `HERE BOV Jump`

执行指令前
PC = 地址 （HERE）
执行指令后
如果溢出标志位 = 1；
PC = 地址 （Jump）
如果溢出标志位 = 0；
PC = 地址 （HERE + 2）

BZ 为零则跳转

语法： BZ n

操作数： -128 ≤ n ≤ 127

操作： 如果全零标志位为 1
(PC) + 2 + 2n → PC

受影响的状态位： 无

机器码：

1110	0000	nnnn	nnnn

说明： 如果全零标志位为 1，那么程序将跳转。二进制补码 “2n”与 PC 相加。因为 PC 要先递增以便取出下一条指令，所以新地址将为 PC + 2 + 2n。这种情况下，该指令是一条双周期指令。

指令字数： 1

指令周期数： 1 （2）

Q 周期操作：

如果跳转：

Q1	Q2	Q3	Q4
译码	读立即数 n	处理数据	写入 PC
空操作	空操作	空操作	空操作

如果不跳转：

Q1	Q2	Q3	Q4
译码	读立即数 n	处理数据	空操作

示例： HERE BZ Jump

执行指令前
PC = 地址 （HERE）
执行指令后
如果全零标志位= 1；
PC = 地址 （Jump）
如果全零标志位= 0；
PC = 地址 （HERE + 2）

CALL 调用子程序

语法： CALL k {,s}

操作数： 0 ≤ k ≤ 1048575
s ∈ [0,1]

操作： (PC)+4 → TOS，
k → PC<20:1>，
如果 s = 1
(W) → WS，
(Status) → STATUSS，
(BSR) → BSRS

受影响的状态位： 无

机器码：

第一个字 （k<7:0>）	1110	110s	$k_7$kkk	kkkk$_0$
第二个字 （k<19:8>）	1111	k_{19}kkk	kkkk	kkkk$_8$

说明： 可在整个 2 MB 的存储器范围内进行子程序调用。首先，将返回地址 （PC+4）压入返回堆栈。如果 s = 1，还会将 W、Status 和 BSR 寄存器的内容存入它们各自的影子寄存器 WS、 STATUSS 和 BSRS。如果 s = 0，将不会进行任何更新 （默认）。然后将 “k”的 20 位值装入 PC<20:1>。 CALL 是一条双周期指令。

指令字数： 2

指令周期数： 2

Q 周期操作：

Q1	Q2	Q3	Q4
译码	读立即数 k<7:0>	将 PC 压入堆栈	读立即数 k<19:8>，写入 PC
空操作	空操作	空操作	空操作

示例： HERE CALL THERE,1

执行指令前
PC = 地址 （HERE）
执行指令后
PC = 地址 （THERE）
TOS = 地址 （HERE + 4）
WS = W
BSRS = BSR
STATUSS = Status

CLRF **将 f 清零**

语法：	CLRF f {,a}
操作数：	$0 \leqslant f \leqslant 255$ $a \in [0,1]$
操作：	000h → f 1 → Z
受影响的状态位：	Z
机器码：	`0110` `101a` `ffff` `ffff`
说明：	清零指定寄存器的内容。 如果 a 为 `0`，选择快速操作存储区。如果 a 为 `1`，使用 BSR 选择 GPR 存储区（默认）。 如果 a 为 `0` 且使能了扩展的指令集，只要 $f \leqslant 95$（5Fh），指令就将以立即数变址寻址模式进行操作。详情请参见**第 26.2.3 节“立即数变址寻址模式中面向字节和位的指令”**。
指令字数：	1
指令周期数：	1

Q 周期操作：

Q1	Q2	Q3	Q4
译码	读寄存器 f	处理数据	写寄存器 f

示例： `CLRF FLAG_REG,1`

执行指令前
FLAG_REG = 5Ah
执行指令后
FLAG_REG = 00h

CLRWDT **将看门狗定时器清零**

语法：	CLRWDT
操作数：	无
操作：	000h → WDT， 000h → WDT 后分频器， 1 → $\overline{TO}$， 1 → $\overline{PD}$
受影响的状态位：	$\overline{TO}$ 和 $\overline{PD}$
机器码：	`0000` `0000` `0000` `0100`
说明：	`CLRWDT` 指令复位看门狗定时器，而且还会复位 WDT 的后分频器。状态位 $\overline{TO}$ 和 $\overline{PD}$ 被置 1。
指令字数：	1
指令周期数：	1

Q 周期操作：

Q1	Q2	Q3	Q4
译码	空操作	处理数据	空操作

示例： `CLRWDT`

执行指令前
WDT 计数器 = ?
执行指令后
WDT 计数器 = 00h
WDT 后分频器 = 0
$\overline{TO}$ = 1
$\overline{PD}$ = 1

COMF **将 f 取反**

语法： COMF f {,d {,a}}

操作数： 0 ≤ f ≤ 255
d ∈ [0,1]
a ∈ [0,1]

操作： $(\bar{f}) \rightarrow$ dest

受影响的状态位： N 和 Z

机器码：

0001	11da	ffff	ffff

说明： 将寄存器 f 的内容取反。如果 d 为 0，结果存储在 W 中。如果 d 为 1，结果存回寄存器 f （默认）。
如果 a 为 0，选择快速操作存储区。如果 a 为 1，使用 BSR 选择 GPR 存储区（默认）。
如果 a 为 0 且使能了扩展的指令集，只要 f ≤ 95 （5Fh），指令就将以立即数变址寻址模式进行操作。详情请参见**第 26.2.3 节“立即数变址寻址模式中面向字节和位的指令”**。

指令字数： 1

指令周期数： 1

Q 周期操作：

Q1	Q2	Q3	Q4
译码	读 寄存器 f	处理数据	写入 目标寄存器

示例： `COMF REG, 0, 0`

执行指令前
REG = 13h
执行指令后
REG = 13h
W = ECh

CPFSEQ **比较 f 和 W，如果 f = W 则跳过**

语法： CPFSEQ f {,a}

操作数： 0 ≤ f ≤ 255
a ∈ [0,1]

操作： (f) – (W)，
如果 (f)=(W) 则跳过
（无符号比较）

受影响的状态位： 无

机器码：

0110	001a	ffff	ffff

说明： 通过执行无符号减法，将数据存储单元 f 的内容与 W 的内容作比较。
如果 f = W，则所取的指令被丢弃，转而执行一条 NOP 指令，从而使该指令变成双周期指令。
如果 a 为 0，选择快速操作存储区。如果 a 为 1，使用 BSR 选择 GPR 存储区（默认）。
如果 a 为 0 且使能了扩展的指令集，只要 f ≤ 95 （5Fh），指令就将以立即数变址寻址模式进行操作。详情请参见**第 26.2.3 节“立即数变址寻址模式中面向字节和位的指令”**。

指令字数： 1

指令周期数： 1 （2）
注：如果跳过的指令后面跟有双字指令，则执行该指令需要 3 个周期。

Q 周期操作：

Q1	Q2	Q3	Q4
译码	读 寄存器 f	处理数据	空操作

如果跳过：

Q1	Q2	Q3	Q4
空操作	空操作	空操作	空操作

如果跳过的指令后面跟有双字指令：

Q1	Q2	Q3	Q4
空操作	空操作	空操作	空操作
空操作	空操作	空操作	空操作

示例：

```
HERE    CPFSEQ REG, 0
NEQUAL  :
EQUAL   :
```

执行指令前
PC 地址 = HERE
W = ?
REG = ?
执行指令后
如果 REG = W；
PC = 地址 （EQUAL）
如果 REG ≠ W；
PC = 地址 （NEQUAL）

 初稿

CPFSGT 比较 f 和 W，如果 f > W 则跳过

语法： CPFSGT f {,a}

操作数： 0 ≤ f ≤ 255
a ∈ [0,1]

操作： (f) – (W)，
如果 (f) > (W) 则跳过
（无符号比较）

受影响的状态位： 无

机器码：

0110	010a	ffff	ffff

说明： 通过执行无符号减法，将数据存储单元 f 的内容与 W 的内容作比较。
如果 f 中的数值大于 WREG 中的数值，则所取的指令会被丢弃，转而执行一条 NOP 指令，从而使该指令变成双周期指令。
如果 a 为 0，选择快速操作存储区。如果 a 为 1，使用 BSR 选择 GPR 存储区（默认）。
如果 a 为 0 且使能了扩展的指令集，只要 f ≤ 95 （5Fh），指令就将以立即数变址寻址模式进行操作。详情请参见**第 26.2.3 节“立即数变址寻址模式中面向字节和位的指令”**。

指令字数： 1

指令周期数： 1 （2）
注： 如果跳过的指令后面跟有双字指令，则执行该指令需要 3 个周期。

Q 周期操作：

Q1	Q2	Q3	Q4
译码	读寄存器 f	处理数据	空操作

如果跳过：

Q1	Q2	Q3	Q4
空操作	空操作	空操作	空操作

如果跳过的指令后面跟有双字指令：

Q1	Q2	Q3	Q4
空操作	空操作	空操作	空操作
空操作	空操作	空操作	空操作

示例：

```
HERE       CPFSGT REG, 0
NGREATER   :
GREATER    :
```

执行指令前
PC = 地址 （HERE）
W = ?

执行指令后
如果 REG > W；
PC = 地址 （GREATER）
如果 REG ≤ W；
PC = 地址 （NGREATER）

CPFSLT 比较 f 和 W，如果 f< W 则跳过

语法： CPFSLT f {,a}

操作数： 0 ≤ f ≤ 255
a ∈ [0,1]

操作： (f) – (W)，
如果 (f) < (W) 则跳过
（无符号比较）

受影响的状态位： 无

机器码：

0110	000a	ffff	ffff

说明： 通过执行无符号减法，将数据存储单元 f 的内容与 W 的内容作比较。
如果 f 中的数值小于 W 中的数值，则所取的指令被丢弃，转而执行一条 NOP 指令，使该指令变成双周期指令。
如果 a 为 0，选择快速操作存储区。如果 a 为 1，使用 BSR 选择 GPR 存储区（默认）。

指令字数： 1

指令周期数： 1 （2）
注： 如果跳过的指令后面跟有双字指令，则执行该指令需要 3 个周期。

Q 周期操作：

Q1	Q2	Q3	Q4
译码	读寄存器 f	处理数据	空操作

如果跳过：

Q1	Q2	Q3	Q4
空操作	空操作	空操作	空操作

如果跳过的指令后面跟有双字指令：

Q1	Q2	Q3	Q4
空操作	空操作	空操作	空操作
空操作	空操作	空操作	空操作

示例：

```
HERE    CPFSLT REG, 1
NLESS   :
LESS    :
```

执行指令前
PC = 地址 （HERE）
W = ?

执行指令后
如果 REG < W；
PC = 地址 （LESS）
如果 REG ≥ W；
PC = 地址 （NLESS）

DAW　对 W 寄存器进行十进制调整

语法：　DAW

操作数：　无

操作：　如果 [W<3:0> > 9] 或 [DC = 1] 那么
(W<3:0>) + 6 → W<3:0>；
否则
(W<3:0>) → W<3:0>

如果 [W<7:4> + DC > 9] 或 [C = 1] 那么
(W<7:4>) + 6 + DC → W<7:4>；
否则
(W<7:4>) + DC → W<7:4>

受影响的状态位：　C

机器码：

0000	0000	0000	0111

说明：　DAW 指令调整 W 内的 8 位数值，即前两个压缩 BCD 格式的变量之和，并产生一个正确的压缩 BCD 格式结果。

指令字数：　1

指令周期数：　1

Q 周期操作：

Q1	Q2	Q3	Q4
译码	读 寄存器 W	处理数据	写 寄存器 W

例 1：　DAW

执行指令前
W = A5h
C = 0
DC = 0
执行指令后
W = 05h
C = 1
DC = 0

例 2：

执行指令前
W = CEh
C = 0
DC = 0
执行指令后
W = 34h
C = 1
DC = 0

DECF　f 减 1

语法：　DECF　f {,d {,a}}

操作数：　0 ≤ f ≤ 255
d ∈ [0,1]
a ∈ [0,1]

操作：　(f) – 1 → dest

受影响的状态位：　C、 DC、 N、 OV 和 Z

机器码：

0000	01da	ffff	ffff

说明：　将寄存器 f 的内容减 1。如果 d 为 0，结果存储在 W 中。如果 d 为 1，结果存回寄存器 f （默认）。
如果 a 为 0，选择快速操作存储区。如果 a 为 1，使用 BSR 选择 GPR 存储区（默认）。
如果 a 为 0 且使能了扩展的指令集，只要 f ≤ 95 （5Fh），指令就将以立即数变址寻址模式进行操作。详情请参见**第 26.2.3 节 "立即数变址寻址模式中面向字节和位的指令"**。

指令字数：　1

指令周期数：　1

Q 周期操作：

Q1	Q2	Q3	Q4
译码	读 寄存器 f	处理数据	写入 目标寄存器

示例：　DECF　CNT,　1, 0

执行指令前
CNT = 01h
Z = 0
执行指令后
CNT = 00h
Z = 1

DECFSZ **f 减 1，为 0 则跳过**

语法：	DECFSZ f {,d {,a}}
操作数：	$0 \leqslant f \leqslant 255$ $d \in [0,1]$ $a \in [0,1]$
操作：	(f) – 1 → dest， 结果为 0 时跳过
受影响的状态位：	无
机器码：	0010 11da ffff ffff
说明：	将寄存器 f 的内容减 1。如果 d 为 0，结果存储在 W 中。如果 d 为 1，结果存回寄存器 f（默认）。 如果结果为 0，则丢弃已取的下一条指令，转而执行一条 NOP 指令，使该指令成为双周期指令。 如果 a 为 0，选择快速操作存储区。如果 a 为 1，使用 BSR 选择 GPR 存储区（默认）。 如果 a 为 0 且使能了扩展的指令集，只要 f ≤ 95（5Fh），指令就将以立即数变址寻址模式进行操作。详情请参见**第 26.2.3 节 “立即数变址寻址模式中面向字节和位的指令”**。
指令字数：	1
指令周期数：	1（2） **注：** 如果跳过的指令后面跟有双字指令，则执行该指令需要 3 个周期。

Q 周期操作：

Q1	Q2	Q3	Q4
译码	读寄存器 f	处理数据	写入目标寄存器

如果跳过：

Q1	Q2	Q3	Q4
空操作	空操作	空操作	空操作

如果跳过的指令后面跟有双字指令：

Q1	Q2	Q3	Q4
空操作	空操作	空操作	空操作
空操作	空操作	空操作	空操作

示例：

```
HERE      DECFSZ   CNT, 1, 1
          GOTO     LOOP
CONTINUE
```

执行指令前
PC = 地址 （HERE）
执行指令后
CNT = CNT–1
如果 CNT = 0；
PC = 地址 （CONTINUE）
如果 CNT ≠ 0；
PC = 地址 （HERE + 2）

DCFSNZ **f 减 1，非 0 则跳过**

语法：	DCFSNZ f {,d {,a}}
操作数：	$0 \leqslant f \leqslant 255$ $d \in [0,1]$ $a \in [0,1]$
操作：	(f) – 1 → dest， 结果 ≠0 时跳过
受影响的状态位：	无
机器码：	0100 11da ffff ffff
说明：	将寄存器 f 的内容减 1。如果 d 为 0，结果存储在 W 中。如果 d 为 1，结果存回寄存器 f（默认）。 如果结果不为 0，则丢弃已经取到的下一条指令，转而执行一条 NOP 指令，使该指令成为双周期指令。 如果 a 为 0，选择快速操作存储区。如果 a 为 1，使用 BSR 选择 GPR 存储区（默认）。 如果 a 为 0 且使能了扩展的指令集，只要 f ≤ 95（5Fh），指令就将以立即数变址寻址模式进行操作。详情请参见**第 26.2.3 节 “立即数变址寻址模式中面向字节和位的指令”**。
指令字数：	1
指令周期数：	1（2） **注：** 如果跳过的指令后面跟有双字指令，则执行该指令需要 3 个周期。

Q 周期操作：

Q1	Q2	Q3	Q4
译码	读寄存器 f	处理数据	写入目标寄存器

如果跳过：

Q1	Q2	Q3	Q4
空操作	空操作	空操作	空操作

如果跳过的指令后面跟有双字指令：

Q1	Q2	Q3	Q4
空操作	空操作	空操作	空操作
空操作	空操作	空操作	空操作

示例：

```
HERE    DCFSNZ  TEMP, 1, 0
ZERO    :
NZERO   :
```

执行指令前
TEMP = ?
执行指令后
TEMP = TEMP-1，
如果 TEMP = 0；
PC = 地址 （ZERO）
如果 TEMP ≠ 0；
PC = 地址 （NZERO）

GOTO 无条件跳转

语法：	GOTO k
操作数：	0 ≤ k ≤ 1048575
操作：	k → PC<20:1>
受影响的状态位：	无

机器码：

第一个字 （k<7:0>）	1110	1111	$k_7$kkk	kkkk$_0$
第二个字 （k<19:8>）	1111	k_{19}kkk	kkkk	kkkk$_8$

说明： GOTO 指令允许无条件跳转到整个 2 MB 存储器范围中的任何位置。将 20 位值 k 装入 PC<20:1>。 GOTO 始终为一条双周期指令。

指令字数： 2

指令周期数： 2

Q 周期操作：

Q1	Q2	Q3	Q4
译码	读立即数 k<7:0>	空操作	读立即数 k<19:8>，写入 PC
空操作	空操作	空操作	空操作

示例： `GOTO THERE`

执行指令后
PC = 地址 （THERE）

INCF f 加 1

语法：	INCF f {,d {,a}}
操作数：	0 ≤ f ≤ 255 d ∈ [0,1] a ∈ [0,1]
操作：	(f) + 1 → dest
受影响的状态位：	C、 DC、 N、 OV 和 Z

机器码：

0010	10da	ffff	ffff

说明： 将寄存器 f 的内容加 1。如果 d 为 0，结果存储在 W 中。如果 d 为 1，结果存回寄存器 f （默认）。
如果 a 为 0，选择快速操作存储区。如果 a 为 1，使用 BSR 选择 GPR 存储区（默认）。
如果 a 为 0 且使能了扩展的指令集，只要 f ≤ 95 （5Fh），指令就将以立即数变址寻址模式进行操作。详情请参见**第 26.2.3 节“立即数变址寻址模式中面向字节和位的指令”**。

指令字数： 1

指令周期数： 1

Q 周期操作：

Q1	Q2	Q3	Q4
译码	读寄存器 f	处理数据	写入目标寄存器

示例： `INCF CNT, 1, 0`

执行指令前
CNT = FFh
Z = 0
C = ?
DC = ?

执行指令后
CNT = 00h
Z = 1
C = 1
DC = 1

INCFSZ　f 加 1，为 0 则跳过

语法：	INCFSZ　f {,d {,a}}
操作数：	0 ≤ f ≤ 255 d ∈ [0,1] a ∈ [0,1]
操作：	(f) + 1 → dest 结果为 0 时跳过
受影响的状态位：	无
机器码：	0011　11da　ffff　ffff
说明：	将寄存器 f 的内容加 1。如果 d 为 0，结果存储在 W 中。如果 d 为 1，结果存回寄存器 f （默认）。 如果结果为 0，则丢弃已取的下一条指令，转而执行一条 NOP 指令，使该指令成为双周期指令。 如果 a 为 0，选择快速操作存储区。如果 a 为 1，使用 BSR 选择 GPR 存储区（默认）。 如果 a 为 0 且使能了扩展的指令集，只要 f ≤ 95 （5Fh），指令就将以立即数变址寻址模式进行操作。详情请参见**第 26.2.3 节"立即数变址寻址模式中面向字节和位的指令"**。
指令字数：	1
指令周期数：	1 （2） **注：** 如果跳过的指令后面跟有双字指令，则执行该指令需要 3 个周期。

Q 周期操作：

Q1	Q2	Q3	Q4
译码	读寄存器 f	处理数据	写入目标寄存器

如果跳过：

Q1	Q2	Q3	Q4
空操作	空操作	空操作	空操作

如果跳过的指令后面跟有双字指令：

Q1	Q2	Q3	Q4
空操作	空操作	空操作	空操作
空操作	空操作	空操作	空操作

示例：

```
HERE     INCFSZ   CNT, 1, 0
NZERO    :
ZERO     :
```

执行指令前
　PC = 地址 （HERE）
执行指令后
　CNT = CNT + 1
　如果 CNT = 0；
　PC = 地址 （ZERO）
　如果 CNT ≠ 0；
　PC = 地址 （NZERO）

INFSNZ　f 加 1，非 0 则跳过

语法：	INFSNZ　f {,d {,a}}
操作数：	0 ≤ f ≤ 255 d ∈ [0,1] a ∈ [0,1]
操作：	(f) + 1 → dest 结果 ≠ 0 时跳过
受影响的状态位：	无
机器码：	0011　11da　ffff　ffff
说明：	将寄存器 f 的内容加 1。如果 d 为 0，结果存储在 W 中。如果 d 为 1，结果存回寄存器 f （默认）。 如果结果不为 0，则丢弃已经取到的下一条指令，转而执行一条 NOP 指令，使该指令成为双周期指令。 如果 a 为 0，选择快速操作存储区。如果 a 为 1，使用 BSR 选择 GPR 存储区（默认）。 如果 a 为 0 且使能了扩展的指令集，只要 f ≤ 95 （5Fh），指令就将以立即数变址寻址模式进行操作。详情请参见**第 26.2.3 节"立即数变址寻址模式中面向字节和位的指令"**。
指令字数：	1
指令周期数：	1 （2） **注：** 如果跳过的指令后面跟有双字指令，则执行该指令需要 3 个周期。

Q 周期操作：

Q1	Q2	Q3	Q4
译码	读寄存器 f	处理数据	写入目标寄存器

如果跳过：

Q1	Q2	Q3	Q4
空操作	空操作	空操作	空操作

如果跳过的指令后面跟有双字指令：

Q1	Q2	Q3	Q4
空操作	空操作	空操作	空操作
空操作	空操作	空操作	空操作

示例：

```
HERE     INFSNZ  REG, 1, 0
ZERO
NZERO
```

执行指令前
　PC = 地址 （HERE）
执行指令后
　REG = REG + 1
　如果 REG ≠ 0；
　PC = 地址 （NZERO）
　如果 REG = 0；
　PC = 地址 （ZERO）

IORLW **立即数与 W 作逻辑或运算**

语法：	IORLW k
操作数：	0 ≤ k ≤ 255
操作：	(W) .OR. k → W
受影响的状态位：	N 和 Z
机器码：	0000 1001 kkkk kkkk
说明：	将 W 的内容与 8 位立即数 k 作逻辑或运算。结果保存在 W 寄存器中。
指令字数：	1
指令周期数：	1

Q 周期操作：

Q1	Q2	Q3	Q4
译码	读 立即数 k	处理数据	写入 寄存器 W

示例： IORLW 35h

执行指令前
W = 9Ah
执行指令后
W = BFh

IORWF **将 W 与 f 作逻辑或运算**

语法：	IORWF f {,d {,a}}
操作数：	0 ≤ f ≤ 255 d ∈ [0,1] a ∈ [0,1]
操作：	(W) .OR. (f) → dest
受影响的状态位：	N 和 Z
机器码：	0001 00da ffff ffff
说明：	将 W 与寄存器 f 的内容作逻辑或运算。如果 d 为 0，结果存储在 W 中。如果 d 为 1，结果存回寄存器 f （默认）。 如果 a 为 0，选择快速操作存储区。如果 a 为 1，使用 BSR 选择 GPR 存储区（默认）。 如果 a 为 0 且使能了扩展的指令集，只要 f ≤ 95 （5Fh），指令就将以立即数变址寻址模式进行操作。详情请参见**第 26.2.3 节“立即数变址寻址模式中面向字节和位的指令”**。
指令字数：	1
指令周期数：	1

Q 周期操作：

Q1	Q2	Q3	Q4
译码	读 寄存器 f	处理数据	写入 目标寄存器

示例： IORWF RESULT, 0, 1

执行指令前
RESULT = 13h
W = 91h
执行指令后
RESULT = 13h
W = 93h

LFSR **装入 FSR**

语法： LFSR f, k

操作数： 0 ≤ f ≤ 2
0 ≤ k ≤ 4095

操作： k → FSRf

受影响的状态位： 无

机器码：

1110	1110	00ff	k_{11}kkk
1111	0000	$k_7$kkk	kkkk

说明： 将 12 位立即数 k 装入由 f 指向的指针寄存器。

指令字数： 2

指令周期数： 2

Q 周期操作：

Q1	Q2	Q3	Q4
译码	读立即数 k 的 MSB	处理数据	写立即数 k 的 MSB 到 FSRfH
译码	读立即数 k 的 LSB	处理数据	将立即数 k 写入 FSRfL

示例： `LFSR 2, 3ABh`

执行指令后
FSR2H = 03h
FSR2L = ABh

MOVF **传送 f**

语法： MOVF f {,d {,a}}

操作数： 0 ≤ f ≤ 255
d ∈ [0,1]
a ∈ [0,1]

操作： f → dest

受影响的状态位： N 和 Z

机器码：

0101	00da	ffff	ffff

说明： 根据 d 的状态，将寄存器 f 的内容送入目标单元。如果 d 为 `0`，结果存储在 W 中。如果 d 为 `1`，结果存回寄存器 f （默认）。 f 可以为 256 字节存储区中的任何单元。
如果 a 为 `0`，选择快速操作存储区。如果 a 为 `1`，使用 BSR 选择 GPR 存储区（默认）。
如果 a 为 `0` 且使能了扩展的指令集，只要 f ≤ 95 （5Fh），指令就将以立即数变址寻址模式进行操作。详情请参见**第 26.2.3 节“立即数变址寻址模式中面向字节和位的指令”**。

指令字数： 1

指令周期数： 1

Q 周期操作：

Q1	Q2	Q3	Q4
译码	读寄存器 f	处理数据	写 W

示例： `MOVF REG, 0, 0`

执行指令前
REG = 22h
W = FFh

执行指令后
REG = 22h
W = 22h

MOVFF　将源寄存器的内容送入目标寄存器

语法：　MOVFF f_s,f_d

操作数：　$0 \leqslant f_s \leqslant 4095$
$0 \leqslant f_d \leqslant 4095$

操作：　$(f_s) \rightarrow f_d$

受影响的状态位：　无

机器码：

第一个字 （源）	1100	ffff	ffff	$ffff_s$
第二个字 （目标）	1111	ffff	ffff	$ffff_d$

说明：　将源寄存器 f_s 的内容送入目标寄存器 f_d。
源寄存器 f_s 可以是 4096 字节数据空间（000h 到 FFFh）中的任何地址，目标寄存器 f_d 也可以是 000h 到 FFFh 中的任何地址。
源或目标寄存器都可以是 W （这是个有用的特例）。
MOVFF 指令对于将数据存储单元中的内容送入外设寄存器 （如发送缓冲器或 I/O 端口）的场合非常有用。
MOVFF 指令中不能使用 PCL、 TOSU、 TOSH 或 TOSL 作为目标寄存器。

指令字数：　2

指令周期数：　2

Q 周期操作：

Q1	Q2	Q3	Q4
译码	读寄存器 f（源寄存器）	处理数据	空操作
译码	空操作 非无效读取	空操作	写 目标寄存器 f

示例：　MOVFF REG1, REG2

执行指令前
REG1 = 33h
REG2 = 11h
执行指令后
REG1 = 33h
REG2 = 33h

MOVLB　将立即数送入 BSR 的低半字节

语法：　MOVLW k

操作数：　$0 \leqslant k \leqslant 255$

操作：　$k \rightarrow$ BSR

受影响的状态位：　无

机器码：

0000	0001	kkkk	kkkk

说明：　将 8 位立即数 k 装入存储区选择寄存器（BSR）。不管 k_7:k_4 的值如何，BSR<7:4> 的值将始终保持为 0。

指令字数：　1

指令周期数：　1

Q 周期操作：

Q1	Q2	Q3	Q4
译码	读 立即数 k	处理数据	将立即数 k 写入 BSR

示例：　MOVLB 5

执行指令前
BSR 寄存器 = 02h
执行指令后
BSR 寄存器 = 05h

MOVLW　　将立即数送入 W

语法：	MOVLW k
操作数：	0 ≤ k ≤ 255
操作：	k → W
受影响的状态位：	无
机器码：	0000 1110 kkkk kkkk
说明：	将 8 位立即数 k 装入 W。
指令字数：	1
指令周期数：	1

Q 周期操作：

Q1	Q2	Q3	Q4
译码	读 立即数 k	处理数据	写入寄存器 W

示例：　　`MOVLW 5Ah`

执行指令后
W = 5Ah

MOVWF　　将 W 的内容送入 f

语法：	MOVWF f {,a}
操作数：	0 ≤ f ≤ 255 a ∈ [0,1]
操作：	(W) → f
受影响的状态位：	无
机器码：	0110 111a ffff ffff
说明：	将 W 寄存器中的数据送入寄存器 f。 f 可以是 256 字节存储区中的任何单元。 如果 a 为 0，选择快速操作存储区。如果 a 为 1，使用 BSR 选择 GPR 存储区（默认）。 如果 a 为 0 且使能了扩展的指令集，只要 f ≤ 95 （5Fh），指令就将以立即数变址寻址模式进行操作。详情请参见**第 26.2.3 节 “立即数变址寻址模式中面向字节和位的指令”**。
指令字数：	1
指令周期数：	1

Q 周期操作：

Q1	Q2	Q3	Q4
译码	读 寄存器 f	处理数据	写 寄存器 f

示例：　　`MOVWF REG, 0`

执行指令前
W = 4Fh
REG = FFh
执行指令后
W = 4Fh
REG = 4Fh

MULLW 将立即数与 W 的内容相乘

语法： MULLW k

操作数： 0 ≤ k ≤ 255

操作： (W) x k → PRODH:PRODL

受影响的状态位： 无

机器码：

0000	1101	kkkk	kkkk

说明： 将 W 的内容与 8 位立即数 k 进行无符号的乘法运算。 16 位的结果存储在 PRODH:PRODL 寄存器对中。其中 PRODH 存储高字节。
W 的内容不变。
所有状态标志位都不受影响。
请注意此操作不可能发生溢出或进位。结果有可能为零，但不会反映到相应的标志位。

指令字数： 1

指令周期数： 1

Q 周期操作：

Q1	Q2	Q3	Q4
译码	读 立即数 k	处理数据	写寄存器 PRODH: PRODL

示例： `MULLW 0C4h`

执行指令前
- W = E2h
- PRODH = ?
- PRODL = ?

执行指令后
- W = E2h
- PRODH = ADh
- PRODL = 08h

MULWF 将 W 与 f 的内容相乘

语法： MULWF f {,a}

操作数： 0 ≤ f ≤ 255
a ∈ [0,1]

操作： (W) x (f) → PRODH:PRODL

受影响的状态位： 无

机器码：

0000	001a	ffff	ffff

说明： 将 W 的内容与寄存器单元 f 的内容执行无符号的乘法运算。运算的 16 位结果存储在 PRODH:PRODL 寄存器对中。其中 PRODH 存储高字节。 W 和 f 的内容都不变。
所有状态标志位都不受影响。
请注意此操作不可能发生溢出或进位。结果有可能为零，但不会反映到相应的标志位。
如果 a 为 0，选择快速操作存储区。如果 a 为 1，使用 BSR 选择 GPR 存储区 （默认）。
如果 a 为 0 且使能了扩展的指令集，只要 f ≤ 95 （5Fh），指令就将以立即数变址寻址模式进行操作。详情请参见第 **26.2.3 节 “立即数变址寻址模式中面向字节和位的指令”**。

指令字数： 1

指令周期数： 1

Q 周期操作：

Q1	Q2	Q3	Q4
译码	读 寄存器 f	处理数据	写寄存器 PRODH: PRODL

示例： `MULWF REG, 1`

执行指令前
- W = C4h
- REG = B5h
- PRODH = ?
- PRODL = ?

执行指令后
- W = C4h
- REG = B5h
- PRODH = 8Ah
- PRODL = 94h

 初稿

NEGF **对 f 取补**

语法： NEGF f {,a}

操作数： 0 ≤ f ≤ 255
a ∈ [0,1]

操作： $(\overline{f}) + 1 \rightarrow f$

受影响的状态位： N、 OV、 C、 DC 和 Z

机器码：

0110	110a	ffff	ffff

说明： 用二进制补码对单元 f 取补。结果存储在数据存储单元 f 中。
如果 a 为 0，选择快速操作存储区。如果 a 为 1，使用 BSR 选择 GPR 存储区（默认）。
如果 a 为 0 且使能了扩展的指令集，只要 f ≤ 95 （5Fh），指令就将以立即数变址寻址模式进行操作。详情请参见**第 26.2.3 节 “立即数变址寻址模式中面向字节和位的指令”**。

指令字数： 1

指令周期数： 1

Q 周期操作：

Q1	Q2	Q3	Q4
译码	读 寄存器 f	处理数据	写 寄存器 f

示例： NEGF REG, 1

执行指令前
REG = 0011 1010 [3Ah]
执行指令后
REG = 1100 0110 [C6h]

NOP **空操作**

语法： NOP

操作数： 无

操作： 空操作

受影响的状态位： 无

机器码：

0000 1111	0000 xxxx	0000 xxxx	0000 xxxx

说明： 不执行任何操作。

指令字数： 1

指令周期数： 1

Q 周期操作：

Q1	Q2	Q3	Q4
译码	空操作	空操作	空操作

示例：

无。

POP 弹出返回堆栈栈顶的内容

语法：	POP
操作数：	无
操作：	(TOS) → 丢弃
受影响的状态位：	无
机器码：	0000 0000 0000 0110
说明：	从返回堆栈弹出 TOS 值并丢弃。然后，前一个压入返回堆栈的值成为 TOS 值。此指令可以让用户正确管理返回堆栈，从而实现软件堆栈。
指令字数：	1
指令周期数：	1

Q 周期操作：

Q1	Q2	Q3	Q4
译码	空操作	弹出 TOS 值	空操作

示例：

```
POP
GOTO      NEW
```

执行指令前

TOS	=	0031A2h
堆栈 （向下 1 级）	=	014332h

执行指令后

TOS	=	014332h
PC	=	NEW

PUSH 将数据压入返回堆栈栈顶

语法：	PUSH
操作数：	无
操作：	(PC + 2) → TOS
受影响的状态位：	无
机器码：	0000 0000 0000 0101
说明：	PC + 2 的值被压入返回堆栈的栈顶。原先的 TOS 值被压入堆栈的下一级。此指令允许通过修改 TOS 并将其压入返回堆栈来实现软件堆栈。
指令字数：	1
指令周期数：	1

Q 周期操作：

Q1	Q2	Q3	Q4
译码	将 PC+2 压入返回堆栈	空操作	空操作

示例：

```
PUSH
```

执行指令前

TOS	=	345Ah
PC	=	0124h

执行指令后

PC	=	0126h
TOS	=	0126h
堆栈 （向下一级）	=	345Ah

初稿

RCALL 相对调用

语法： RCALL n

操作数： -1024 ≤ n ≤ 1023

操作： (PC) + 2 → TOS，
(PC) + 2 + 2n → PC

受影响的状态位： 无

机器码：

1101	1nnn	nnnn	nnnn

说明： 从当前地址跳转 （最多 1 KB）来调用子程序。首先，将返回地址 （PC + 2）压入返回堆栈。然后，将二进制补码 “2n”与 PC 相加。因为 PC 要先递增才能取下一条指令，因此新地址将为 PC + 2 + 2n。该指令为一条双周期指令。

指令字数： 1

指令周期数： 2

Q 周期操作：

Q1	Q2	Q3	Q4
译码	读立即数 n 将 PC 压入堆栈	处理数据	写入 PC
空操作	空操作	空操作	空操作

示例： HERE RCALL Jump

执行指令前
PC = 地址 （HERE）
执行指令后
PC = 地址 （Jump）
TOS = 地址 （HERE + 2）

RESET 复位

语法： RESET

操作数： 无

操作： 将所有受 $\overline{MCLR}$ 复位影响的寄存器和标志位复位。

受影响的状态位： 全部

机器码：

0000	0000	1111	1111

说明： 此指令可提供一种用软件实现 $\overline{MCLR}$ 复位的方法。

指令字数： 1

指令周期数： 1

Q 周期操作：

Q1	Q2	Q3	Q4
译码	开始复位	空操作	空操作

示例： RESET

执行指令后
寄存器 = 复位值
标志位 * = 复位值

RETFIE 从中断返回

语法：	RETFIE {s}
操作数：	s ∈ [0,1]
操作：	(TOS) → PC， 1 → GIE/GIEH 或 PEIE/GIEL， 如果 s = 1 (WS) → W， (STATUSS) → Status 寄存器， (BSRS) → BSR， PCLATU 和 PCLATH 保持不变。
受影响的状态位：	GIE/GIEH 和 PEIE/GIEL。
机器码：	0000 0000 0001 000s
说明：	从中断返回。执行出栈操作，将栈顶（TOS）的内容装入 PC。通过将高或低优先级全局中断允许位置 1，来允许中断。如果 s = 1，则影子寄存器 WS、STATUSS 和 BSRS 的内容将被装入对应的寄存器 W、Status 和 BSR。如果 s = 0，则不更新这些寄存器（默认）。
指令字数：	1
指令周期数：	2

Q 周期操作：

Q1	Q2	Q3	Q4
译码	空操作	空操作	从堆栈弹出 PC 值 将 GIEH 或 GIEL 置 1
空操作	空操作	空操作	空操作

示例： `RETFIE 1`

中断后

PC	=	TOS
W	=	WS
BSR	=	BSRS
Status	=	STATUSS
GIE/GIEH, PEIE/GIEL	=	1

RETLW 返回时将立即数送入 **W**

语法：	RETLW k
操作数：	0 ≤ k ≤ 255
操作：	k → W， (TOS) → PC， PCLATU 和 PCLATH 保持不变。
受影响的状态位：	无
机器码：	0000 1100 kkkk kkkk
说明：	将 8 位立即数 k 装入 W。将栈顶内容（返回地址）装入程序计数器。高位地址锁存器（PCLATH）的内容保持不变。
指令字数：	1
指令周期数：	2

Q 周期操作：

Q1	Q2	Q3	Q4
译码	读立即数 k	处理数据	从堆栈弹出 PC 值，写入 W
空操作	空操作	空操作	空操作

示例：

```
    CALL TABLE ; W contains table
               ; offset value
               ; W now has
               ; table value
   :
TABLE
    ADDWF PCL  ; W = offset
    RETLW k0   ; Begin table
    RETLW k1   ;
   :
   :
    RETLW kn   ; End of table
```

执行指令前
W = 07h
执行指令后
W = kn 的值

RETURN 从子程序返回

语法：	RETURN {s}
操作数：	s ∈ [0,1]
操作：	(TOS) → PC， 如果 s = 1 (WS) → W， (STATUSS) → Status 寄存器， (BSRS) → BSR， PCLATU 和 PCLATH 保持不变。
受影响的状态位：	无
机器码：	0000 | 0000 | 0001 | 001s
说明：	从子程序返回。执行出栈操作，将栈顶（TOS）内容装入程序计数器。如果 s = 1，将影子寄存器 WS、 STATUSS 和 BSRS 的内容装入相应的 W、 Status 和 BSR 寄存器。如果 s = 0，则不更新这些寄存器 （默认）。
指令字数：	1
指令周期数：	2

Q 周期操作：

Q1	Q2	Q3	Q4
译码	空操作	处理数据	从堆栈弹出 PC 值
空操作	空操作	空操作	空操作

示例： RETURN

执行指令后：
PC = TOS

RLCF f 带进位循环左移

语法：	RLCF f {,d {,a}}
操作数：	0 ≤ f ≤ 255 d ∈ [0,1] a ∈ [0,1]
操作：	(f<n>) → dest<n+1>， (f<7>) → C， (C) → dest<0>
受影响的状态位：	C、 N 和 Z
机器码：	0011 | 01da | ffff | ffff
说明：	将寄存器 f 的内容连同进位标志位一起循环左移 1 位。如果 d 为 0，结果存储在 W 中。如果 d 为 1，结果存回寄存器 f （默认）。 如果 a 为 0，选择快速操作存储区。如果 a 为 1，使用 BSR 选择 GPR 存储区 （默认）。 如果 a 为 0 且使能了扩展的指令集，只要 f ≤ 95 （5Fh），指令就将以立即数变址寻址模式进行操作。详情请参见**第 26.2.3 节 "立即数变址寻址模式中面向字节和位的指令"**。 C ← 寄存器 f
指令字数：	1
指令周期数：	1

Q 周期操作：

Q1	Q2	Q3	Q4
译码	读寄存器 f	处理数据	写入目标寄存器

示例： RLCF REG, 0, 0

执行指令前
REG = 1110 0110
C = 0

执行指令后
REG = 1110 0110
W = 1100 1100
C = 1

RLNCF　f 循环左移 （不带进位）

语法：　RLNCF　f {,d {,a}}

操作数：　0 ≤ f ≤ 255
d ∈ [0,1]
a ∈ [0,1]

操作：　(f<n>) → dest<n+1>，
(f<7>) → dest<0>

受影响的状态位：　N 和 Z

机器码：

0100	01da	ffff	ffff

说明：　将寄存器 f 的内容循环左移 1 位。如果 d 为 0，结果存储在 W 中。如果 d 为 1，结果存回寄存器 f（默认）。
如果 a 为 0，选择快速操作存储区。如果 a 为 1，使用 BSR 选择 GPR 存储区（默认）。
如果 a 为 0 且使能了扩展的指令集，只要 f ≤ 95 （5Fh），指令就将以立即数变址寻址模式进行操作。详情请参见**第 26.2.3 节 "立即数变址寻址模式中面向字节和位的指令"**。

寄存器 f

指令字数：　1

指令周期数：　1

Q 周期操作：

Q1	Q2	Q3	Q4
译码	读 寄存器 f	处理数据	写入 目标寄存器

示例：　`RLNCF　REG, 1, 0`

执行指令前
REG = `1010 1011`
执行指令后
REG = `0101 0111`

RRCF　f 带进位循环右移

语法：　RRCF　f {,d {,a}}

操作数：　0 ≤ f ≤ 255
d ∈ [0,1]
a ∈ [0,1]

操作：　(f<n>) → dest<n-1>，
(f<0>) → C，
(C) → dest<7>

受影响的状态位：　C、N 和 Z

机器码：

0011	00da	ffff	ffff

说明：　将寄存器 f 的内容连同进位标志位一起循环右移 1 位。如果 d 为 0，结果存储在 W 中。如果 d 为 1，结果存回寄存器 f（默认）。
如果 a 为 0，选择快速操作存储区。如果 a 为 1，使用 BSR 选择 GPR 存储区（默认）。
如果 a 为 0 且使能了扩展的指令集，只要 f ≤ 95 （5Fh），指令就将以立即数变址寻址模式进行操作。详情请参见**第 26.2.3 节 "立即数变址寻址模式中面向字节和位的指令"**。

C　寄存器 f

指令字数：　1

指令周期数：　1

Q 周期操作：

Q1	Q2	Q3	Q4
译码	读 寄存器 f	处理数据	写入 目标寄存器

示例：　`RRCF　REG, 0, 0`

执行指令前
REG = `1110 0110`
C = `0`
执行指令后
REG = `1110 0110`
W = `0111 0011`
C = `0`

RRNCF　f 循环右移 （不带进位）

语法：	RRNCF　f {,d {,a}}
操作数：	0 ≤ f ≤ 255 d ∈ [0,1] a ∈ [0,1]
操作：	(f<n>) → dest<n-1>， (f<n>) → dest<7>
受影响的状态位：	N 和 Z
机器码：	0100 00da ffff ffff
说明：	将寄存器 f 的内容循环右移 1 位。如果 d 为 0，结果存储在 W 中。如果 d 为 1，结果存回寄存器 f（默认）。 如果 a 为 0，选择快速操作存储区，忽略 BSR 的值。如果 a 为 1，使用 BSR 选择 GPR 存储区 （默认）。 如果 a 为 0 且使能了扩展的指令集，只要 f ≤ 95 （5Fh），指令就将以立即数变址寻址模式进行操作。详情请参见**第 26.2.3 节“立即数变址寻址模式中面向字节和位的指令”**。 寄存器 f
指令字数：	1
指令周期数：	1

Q 周期操作：

Q1	Q2	Q3	Q4
译码	读 寄存器 f	处理数据	写入 目标寄存器

例 1：　RRNCF　REG, 1, 0

执行指令前
　REG = 1101 0111
执行指令后
　REG = 1110 1011

例 2：　RRNCF　REG, 0, 0

执行指令前
　W = ?
　REG = 1101 0111
执行指令后
　W = 1110 1011
　REG = 1101 0111

SETF　将 f 的内容置为全 1

语法：	SETF　f {,a}
操作数：	0 ≤ f ≤ 255 a ∈ [0,1]
操作：	FFh → f
受影响的状态位：	无
机器码：	0110 100a ffff ffff
说明：	将指定寄存器的内容置为 FFh。 如果 a 为 0，选择快速操作存储区。如果 a 为 1，使用 BSR 选择 GPR 存储区（默认）。 如果 a 为 0 且使能了扩展的指令集，只要 f ≤ 95 （5Fh），指令就将以立即数变址寻址模式进行操作。详情请参见**第 26.2.3 节“立即数变址寻址模式中面向字节和位的指令”**。
指令字数：	1
指令周期数：	1

Q 周期操作：

Q1	Q2	Q3	Q4
译码	读 寄存器 f	处理数据	写 寄存器 f

示例：　SETF　REG,1

执行指令前
　REG = 5Ah
执行指令后
　REG = FFh

SLEEP 进入休眠模式

语法：	SLEEP
操作数：	无
操作：	00h → WDT， 0 → WDT 后分频器， 1 → $\overline{TO}$， 0 → $\overline{PD}$
受影响的状态位：	$\overline{TO}$ 和 $\overline{PD}$
机器码：	0000 0000 0000 0011
说明：	掉电状态位（$\overline{PD}$）清零。超时状态位（$\overline{TO}$）置 1。看门狗定时器及其后分频器清零。 振荡器停振，处理器进入休眠模式。
指令字数：	1
指令周期数：	1

Q 周期操作：

Q1	Q2	Q3	Q4
译码	空操作	处理数据	进入休眠模式

示例： SLEEP

执行指令前
$\overline{TO}$ = ?
$\overline{PD}$ = ?

执行指令后
$\overline{TO}$ = 1 †
$\overline{PD}$ = 0

† 如果由 WDT 引起唤醒，则此位将被清零。

SUBFWB W 减去 f （带借位）

语法：	SUBFWB f {,d {,a}}
操作数：	0 ≤ f ≤ 255 d ∈ [0,1] a ∈ [0,1]
操作：	(W) – (f) – ($\overline{C}$) → dest
受影响的状态位：	N、 OV、 C、 DC 和 Z
机器码：	0101 01da ffff ffff
说明：	将 W 的内容减去 f 寄存器的内容和进位 （借位）（通过二进制补码方法进行运算）。如果 d 为 0，结果存储在 W 中。如果 d 为 1，结果存回寄存器 f （默认）。 如果 a 为 0，选择快速操作存储区。 如果 a 为 1，使用 BSR 选择 GPR 存储区 （默认）。 如果 a 为 0 且使能了扩展的指令集，只要 f ≤ 95 （5Fh），指令就将以立即数变址寻址模式进行操作。详情请参见第 **26.2.3** 节 **"立即数变址寻址模式中面向字节和位的指令"**。
指令字数：	1
指令周期数：	1

Q 周期操作：

Q1	Q2	Q3	Q4
译码	读寄存器 f	处理数据	写入目标寄存器

例 1： SUBFWB REG, 1, 0

执行指令前
REG = 3
W = 2
C = 1

执行指令后
REG = FF
W = 2
C = 0
Z = 0
N = 1 ；结果为负

例 2： SUBFWB REG, 0, 0

执行指令前
REG = 2
W = 5
C = 1

执行指令后
REG = 2
W = 3
C = 1
Z = 0
N = 0 ；结果为正

例 3： SUBFWB REG, 1, 0

执行指令前
REG = 1
W = 2
C = 0

执行指令后
REG = 0
W = 2
C = 1
Z = 1 ；结果为零
N = 0

SUBLW　立即数减去 W 的内容

语法：SUBLW k

操作数：0 ≤ k ≤ 255

操作：k – (W) → W

受影响的状态位：N、OV、C、DC 和 Z

机器码：

0000	1000	kkkk	kkkk

说明：用 8 位立即数 k 减去 W 的内容。结果保存在 W 寄存器中。

指令字数：1

指令周期数：1

Q 周期操作：

Q1	Q2	Q3	Q4
译码	读 立即数 k	处理数据	写入寄存器 W

例 1：SUBLW 02h

执行指令前
W = 01h
C = ?

执行指令后
W = 01h
C = 1 ；结果为正
Z = 0
N = 0

例 2：SUBLW 02h

执行指令前
W = 02h
C = ?

执行指令后
W = 00h
C = 1 ；结果为零
Z = 1
N = 0

例 3：SUBLW 02h

执行指令前
W = 03h
C = ?

执行指令后
W ＝ FFh ；（2 进制补码）
C = 0 ；结果为负
Z = 0
N = 1

SUBWF　f 减去 W

语法：SUBWF f {,d {,a}}

操作数：0 ≤ f ≤ 255
d ∈ [0,1]
a ∈ [0,1]

操作：(f) – (W) → dest

受影响的状态位：N、OV、C、DC 和 Z

机器码：

0101	11da	ffff	ffff

说明：用寄存器 f 中的内容减去 W 寄存器的内容（通过二进制补码方式进行运算）。如果 d 为 0，结果存储在 W 中。如果 d 为 1，结果存回寄存器 f（默认）。
如果 a 为 0，选择快速操作存储区。
如果 a 为 1，使用 BSR 选择 GPR 存储区（默认）。
如果 a 为 0 且使能了扩展的指令集，只要 f ≤ 95（5Fh），指令就将以立即数变址寻址模式进行操作。详情请参见第 **26.2.3 节 "立即数变址寻址模式中面向字节和位的指令"**。

指令字数：1

指令周期数：1

Q 周期操作：

Q1	Q2	Q3	Q4
译码	读 寄存器 f	处理数据	写入 目标寄存器

例 1：SUBWF REG, 1, 0

执行指令前
REG = 3
W = 2
C = ?

执行指令后
REG = 1
W = 2
C = 1 ；结果为正
Z = 0
N = 0

例 2：SUBWF REG, 0, 0

执行指令前
REG = 2
W = 2
C = ?

执行指令后
REG = 2
W = 0
C = 1 ；结果为零
Z = 1
N = 0

例 3：SUBWF REG, 1, 0

执行指令前
REG = 1
W = 2
C = ?

执行指令后
REG = FFh ；（二进制补码）
W = 2
C = 0 ；结果为负
Z = 0
N = 1

SUBWFB　f 减去 W （带借位）

语法：	SUBWFB　f {,d {,a}}
操作数：	0 ≤ f ≤ 255 d ∈ [0,1] a ∈ [0,1]
操作：	(f) - (W) - ($\overline{C}$) → dest
受影响的状态位：	N、 OV、 C、 DC 和 Z
机器码：	0101 | 10da | ffff | ffff
说明：	用 f 寄存器的内容减去 W 的内容和进位（借位）标志位 （通过二进制补码方式进行运算）。如果 d 为 0，结果存储在 W 中。如果 d 为 1，结果存回寄存器 f（默认）。 如果 a 为 0，选择快速操作存储区。如果 a 为 1，使用 BSR 选择 GPR 存储区（默认）。 如果 a 为 0 且使能了扩展的指令集，只要 f ≤ 95 （5Fh），指令就将以立即数变址寻址模式进行操作。详情请参见 **第 26.2.3 节 "立即数变址寻址模式中面向字节和位的指令"**。
指令字数：	1
指令周期数：	1

Q 周期操作：

Q1	Q2	Q3	Q4
译码	读 寄存器 f	处理数据	写入 目标寄存器

例 1：　SUBWFB　REG, 1, 0

执行指令前
REG = 19h　（0001 1001）
W = 0Dh　（0000 1101）
C = 1

执行指令后
REG = 0Ch　（0000 1011）
W = 0Dh　（0000 1101）
C = 1
Z = 0
N = 0　；结果为正

例 2：　SUBWFB　REG, 0, 0

执行指令前
REG = 1Bh　（0001 1011）
W = 1Ah　（0001 1010）
C = 0

执行指令后
REG = 1Bh　（0001 1011）
W = 00h
C = 1
Z = 1　；结果为零
N = 0

例 3：　SUBWFB　REG, 1, 0

执行指令前
REG = 03h　（0000 0011）
W = 0Eh　（0000 1101）
C = 1

执行指令后
REG = F5h　（1111 0100）
　　；[二进制补码]
W = 0Eh　（0000 1101）
C = 0
Z = 0
N = 1　；结果为负

SWAPF　将 f 的高半字节和低半字节交换

语法：	SWAPF　f {,d {,a}}
操作数：	0 ≤ f ≤ 255 d ∈ [0,1] a ∈ [0,1]
操作：	(f<3:0>) → dest<7:4>, (f<7:4>) → dest<3:0>
受影响的状态位：	无
机器码：	0011 | 10da | ffff | ffff
说明：	将寄存器 f 的高半字节和低半字节互相交换。如果 d 为 0，结果存储在 W 中。如果 d 为 1，结果存回寄存器 f （默认）。 如果 a 为 0，选择快速操作存储区。如果 a 为 1，使用 BSR 选择 GPR 存储区（默认）。 如果 a 为 0 且使能了扩展的指令集，只要 f ≤ 95 （5Fh），指令就将以立即数变址寻址模式进行操作。详情请参见 **第 26.2.3 节 "立即数变址寻址模式中面向字节和位的指令"**。
指令字数：	1
指令周期数：	1

Q 周期操作：

Q1	Q2	Q3	Q4
译码	读 寄存器 f	处理数据	写入 目标寄存器

示例：　SWAPF　REG, 1, 0

执行指令前
REG = 53h

执行指令后
REG = 35h

TBLRD 表读

语法： TBLRD (*; *+; *-; +*)

操作数： 无

操作：
如果执行 TBLRD *，
（程序存储器 (TBLPTR)）→ TABLAT ；
TBLPTR →不改变；
如果执行 TBLRD *+，
（程序存储器 (TBLPTR)）→ TABLAT ；
(TBLPTR) + 1 → TBLPTR ；
如果执行 TBLRD *-，
（程序存储器 (TBLPTR)）→ TABLAT ；
(TBLPTR) –1 → TBLPTR ；
如果执行 TBLRD +*，
(TBLPTR) + 1 → TBLPTR ；
（程序存储器 (TBLPTR)）→ TABLAT ；

受影响的状态位：无

机器码：

0000	0000	0000	10nn nn=0 * =1 *+ =2 *- =3 +*

说明：
该指令用于读取程序存储器 （P.M.）的内容。使用表指针 （TBLPTR）对程序存储器进行寻址。
TBLPTR （一个 21 位指针）指向程序存储器中的每个字节。 TBLPTR 的寻址范围为 2 MB。

TBLPTR[0] = 0： 程序存储器字的低有效字节
TBLPTR[0] = 1： 程序存储器字的高有效字节

TBLRD 指令可用如下方法修改 TBLPTR 的值：
- 不变
- 后加
- 后减
- 预加

指令字数： 1

指令周期数： 2

Q 周期操作：

Q1	Q2	Q3	Q4
译码	空操作	空操作	空操作
空操作	空操作 （读程序存储器）	空操作	空操作 （写 TABLAT）

TBLRD 表读 （续）

例 1： `TBLRD *+ ;`

执行指令前
TABLAT = 55h
TBLPTR = 00A356h
存储单元 （00A356h） = 34h
执行指令后
TABLAT = 34h
TBLPTR = 00A357h

例 2： `TBLRD +* ;`

执行指令前
TABLAT = AAh
TBLPTR = 01A357h
存储单元 （01A357h） = 12h
存储单元 （01A358h） = 34h
执行指令后
TABLAT = 34h
TBLPTR = 01A358h

TBLWT 表写

语法： TBLWT (*; *+; *-; +*)

操作数： 无

操作：
如果执行 TBLWT*，
(TABLAT) →保持寄存器；
TBLPTR →不改变；
如果执行 TBLWT*+，
(TABLAT) →保持寄存器；
(TBLPTR) + 1 → TBLPTR ；
如果执行 TBLWT*-，
(TABLAT) →保持寄存器；
(TBLPTR) –1 → TBLPTR ；
如果执行 TBLWT+*，
(TBLPTR) + 1 → TBLPTR ；
(TABLAT) →保持寄存器；

受影响的状态位：无

机器码：

0000	0000	0000	11nn nn=0 * =1 *+ =2 *- =3 +*

说明：
此指令使用 TBLPTR 的低 3 位来确定要将 TABLAT 中的内容写入 8 个保持寄存器中的哪一个。该保持寄存器用于对程序存储器（P.M.）的内容编程。（关于对闪存存储器编程的更多详情，请参见**第 6.0 节 “闪存程序存储器”**。）
TBLPTR （一个 21 位指针）指向程序存储器中的每个字节。 TBLPTR 的寻址范围为 2 MB。 TBLPTR 的 LSb 选择要访问的程序存储器单元。

TBLPTR[0] = 0： 程序存储器字的低有效字节
TBLPTR[0] = 1： 程序存储器字的高有效字节

TBLWT 指令可用如下方法修改 TBLPTR 的值：
- 不变
- 后加
- 后减
- 预加

指令字数： 1

指令周期数： 2

Q 周期操作：

Q1	Q2	Q3	Q4
译码	空操作	空操作	空操作
空操作	空操作（读 TABLAT）	空操作	空操作（写保持寄存器）

TBLWT 表写 （续）

例 1： TBLWT *+;

执行指令前
TABLAT = 55h
TBLPTR = 00A356h
保持寄存器 (00A356h) = FFh

执行指令后 （表写操作完成）
TABLAT = 55h
TBLPTR = 00A357h
保持寄存器 (00A356h) = 55h

例 2： TBLWT +*;

执行指令前
TABLAT = 34h
TBLPTR = 01389Ah
保持寄存器 (01389Ah) = FFh
保持寄存器 (01389Bh) = FFh

执行指令后 （表写操作完成）
TABLAT = 34h
TBLPTR = 01389Bh
保持寄存器 (01389Ah) = FFh
保持寄存器 (01389Bh) = 34h

TSTFSZ	测试 f，为 0 则跳过
语法：	TSTFSZ f {,a}
操作数：	0 ≤ f ≤ 255 a ∈ [0,1]
操作：	f 为 0 则跳过
受影响的状态位：	无
机器码：	0110 | 011a | ffff | ffff
说明：	如果 f = 0，丢弃已经取到的指令，转而执行一条 NOP 指令，使这条指令成为双周期指令。 如果 a 为 0，选择快速操作存储区。如果 a 为 1，使用 BSR 选择 GPR 存储区（默认）。 如果 a 为 0 且使能了扩展的指令集，只要 f ≤ 95（5Fh），指令就将以立即数变址寻址模式进行操作。详情请参见**第 26.2.3 节 "立即数变址寻址模式中面向字节和位的指令"**。
指令字数：	1
指令周期数：	1（2） **注：**如果跳过的指令后面跟有双字指令，则执行该指令需要 3 个周期。

Q 周期操作：

Q1	Q2	Q3	Q4
译码	读 寄存器 f	处理数据	空操作

如果跳过：

Q1	Q2	Q3	Q4
空操作	空操作	空操作	空操作

如果跳过的指令后面跟有双字指令：

Q1	Q2	Q3	Q4
空操作	空操作	空操作	空操作
空操作	空操作	空操作	空操作

示例：

```
HERE     TSTFSZ  CNT, 1
NZERO    :
ZERO     :
```

执行指令前
PC = 地址（HERE）
执行指令后
如果 CNT = 00h，
PC = 地址（ZERO）
如果 CNT ≠ 00h，
PC = 地址（NZERO）

XORLW	立即数与 W 作逻辑异或运算
语法：	XORLW k
操作数：	0 ≤ k ≤ 255
操作：	(W) .XOR. k → W
受影响的状态位：	N 和 Z
机器码：	0000 | 1010 | kkkk | kkkk
说明：	将 W 的内容与 8 位立即数 k 进行逻辑异或运算。结果保存在 W 寄存器中。
指令字数：	1
指令周期数：	1

Q 周期操作：

Q1	Q2	Q3	Q4
译码	读 立即数 k	处理数据	写入寄存器 W

示例：

```
XORLW   0AFh
```

执行指令前
W = B5h
执行指令后
W = 1Ah

XORWF **W 与 f 作逻辑异或运算**

语法：	XORWF f {,d {,a}}
操作数：	0 ≤ f ≤ 255 d ∈ [0,1] a ∈ [0,1]
操作：	(W) .XOR.(f) → dest
受影响的状态位：	N 和 Z
机器码：	0001 10da ffff ffff
说明：	将 W 的内容与寄存器 f 的内容进行逻辑异或运算。如果 d 为 0，结果存储在 W 中。如果 d 为 1，结果存回寄存器 f （默认）。 如果 a 为 0，选择快速操作存储区。如果 a 为 1，使用 BSR 选择 GPR 存储区（默认）。 如果 a 为 0 且使能了扩展的指令集，只要 f ≤ 95 （5Fh），指令就将以立即数变址寻址模式进行操作。详情请参见**第 26.2.3 节“立即数变址寻址模式中面向字节和位的指令”**。
指令字数：	1
指令周期数：	1

Q 周期操作：

Q1	Q2	Q3	Q4
译码	读 寄存器 f	处理数据	写入 目标寄存器

示例： `XORWF REG, 1, 0`

执行指令前
REG = AFh
W = B5h

执行指令后
REG = 1Ah
W = B5h

 初稿

26.2 扩展的指令集

除了PIC18指令集的75个标准指令之外，PIC18F2455/2550/4455/4550 器件还提供了针对 CPU 内核功能的可选扩展指令。这些新增的功能包括 8 个额外的指令，它们可以实现间接和变址寻址操作；以及为许多标准PIC18 指令实现了立即数变址寻址模式。

这些扩展指令集的额外功能在默认情况下是禁止的。用户必须通过将 XINST 配置位置 1，才能使能它们。

扩展指令集中的指令可以全部被归为立即数操作类指令，它们既可以控制文件选择寄存器，也可以使用这些寄存器进行变址寻址。其中的两条指令 ADDFSR 和SUBFSR，可以直接对 FSR2 进行操作；而 ADDULNK和 SUBULNK 指令允许在执行后自动返回。

这些扩展的指令专门用于优化用高级语言，特别是 C 语言编写的重入程序代码（也就是递归调用或使用软件堆栈的代码）。此外，它们使用户能更有效地用高级语言对数据结构执行特定的操作。这些操作包括：

- 在进入和退出子程序时对软件堆栈空间进行动态分配和释放
- 功能指针调用
- 对软件堆栈指针进行控制
- 对软件堆栈中的变量进行控制

表 26-3 提供了扩展指令集中的指令汇总。第 **26.2.2** 节 **"扩展的指令集"** 对这些指令进行了详细说明。表 26-1（第 302 页）提供了标准和扩展的PIC18 指令集的操作码字段说明。

注：	扩展的指令集和立即数变址寻址模式是专为优化用 C 语言编写的应用程序而设计的，用户可能不会在汇编器中直接使用这些指令。对于那些需要查看编译器生成代码的用户，这些命令的语法可作为参考。

26.2.1 扩展指令的语法

大部分扩展指令都使用变址参数，同时使用一个文件选择寄存器和某一偏移量来指定源寄存器或目标寄存器。当指令的参数作为变址寻址的一部分时，会用方括号（"[ ]"）把它括起来。这时表示此参数用作变址地址或偏移量。如果 MPASM™ 汇编器发现一个变址地址或偏移量没有被括起来，它就会给出错误警告。

当使能扩展的指令集时，括号也用于表示面向字节和面向位的指令中的变址参数。这是对指令语法的额外更改。欲知更多信息，请参见第 **26.2.3.1** 节 **"标准 PIC18 命令的扩展指令语法"**。

注：	以前，在 PIC18 和早期的指令集中使用方括号来表示可选参数。在此文本和以后的文本中，可选参数将用大扩号（"{ }"）表示。

表 26-3： PIC18 指令集的扩展

助记符，操作数		说明	指令周期数	16 位指令字 MSb			LSb	受影响的状态位
ADDFSR	f, k	将立即数与 FSR 相加	1	1110	1000	ffkk	kkkk	无
ADDULNK	k	将立即数与 FSR2 相加并返回	2	1110	1000	11kk	kkkk	无
CALLW		使用 WREG 调用子程序	2	0000	0000	0001	0100	无
MOVSF	z_s, f_d	将 z_s（源）装入第一个字 将 f_d（目标）装入第二个字	2	1110 1111	1011 ffff	0zzz ffff	zzzz ffff	无
MOVSS	z_s, z_d	将 z_s（源）装入第一个字 将 z_d（目标）装入第二个字	2	1110 1111	1011 xxxx	1zzz xzzz	zzzz zzzz	无
PUSHL	k	将立即数保存在 FSR2 后，FSR2 减 1	1	1110	1010	kkkk	kkkk	无
SUBFSR	f, k	FSR 减去立即数	1	1110	1001	ffkk	kkkk	无
SUBULNK	k	FSR2 减去立即数并返回	2	1110	1001	11kk	kkkk	无

26.2.2 扩展的指令集

ADDFSR **FSR 的内容与立即数相加**

语法： ADDFSR f, k

操作数： 0 ≤ k ≤ 63
f ∈ [0, 1, 2]

操作： FSR(f) + k → FSR(f)

受影响的状态位： 无

机器码：

1110	1000	ffkk	kkkk

说明： 将由 f 指定的 FSR 的内容加上一个 6 位的立即数 k。

指令字数： 1

指令周期数： 1

Q 周期操作：

Q1	Q2	Q3	Q4
译码	读 立即数 k	处理数据	写入 FSR

示例： `ADDFSR 2, 23h`

执行指令前
FSR2 = 03FFh

执行指令后
FSR2 = 0422h

ADDULNK **FSR2 的内容与立即数相加并返回**

语法： ADDULNK k

操作数： 0 ≤ k ≤ 63

操作： FSR2 + k → FSR2,
(TOS) → PC

受影响的状态位： 无

机器码：

1110	1000	11kk	kkkk

说明： 将 FSR2 的内容加上一个 6 位的立即数 k。然后通过将 TOS 装入 PC，执行一条 `RETURN` 指令。
执行该指令需要两个周期；在第二个周期执行一条 `NOP` 指令。
该指令可以被认为是 `ADDFSR` 指令的特例，其中 f = 3 （二进制 `11`）；它仅针对 FSR2 进行操作。

指令字数： 1

指令周期数： 2

Q 周期操作：

Q1	Q2	Q3	Q4
译码	读 立即数 k	处理数据	写入 FSR
空操作	空操作	空操作	空操作

示例： `ADDULNK 23h`

执行指令前
FSR2 = 03FFh
PC = 0100h

执行指令后
FSR2 = 0422h
PC = （TOS）

> **注：** 所有的 PIC18 指令都可能在其指令助记符之前使用可选的标号参数，用于符号寻址。如果使用标号，那么指令语法将变为：{ 标号 } 指令参数。

CALLW 使用 WREG 调用子程序

语法：	CALLW
操作数：	无
操作：	（PC + 2）→ TOS， (W) → PCL， (PCLATH) → PCH， (PCLATU) → PCU
受影响的状态位：	无
机器码：	`0000` `0000` `0001` `0100`
说明	首先，返回地址 （PC + 2）被压入返回堆栈。接下来，将 W 寄存器的内容写入 PCL， PCL 现有的值被丢弃。然后， PCLATH 和 PCLATU 的内容被分别锁存到 PCH 和 PCU。第二个周期执行一条 `NOP` 指令，并同时取出下一条指令。 和 `CALL` 不一样，该指令没有更新 W、 Status 或 BSR 寄存器的选项。
指令字数：	1
指令周期数：	2

Q 周期操作：

Q1	Q2	Q3	Q4
译码	读 WREG	将 PC 压入堆栈	空操作
空操作	空操作	空操作	空操作

示例： `HERE CALLW`

执行指令前
- PC = 地址 （HERE）
- PCLATH = 10h
- PCLATU = 00h
- W = 06h

执行指令后
- PC = 001006h
- TOS = 地址 （HERE + 2）
- PCLATH = 10h
- PCLATU = 00h
- W = 06h

MOVSF 将变址寻址单元内容送入 f

语法：	MOVSF [z_s], f_d
操作数：	$0 \leq z_s \leq 127$ $0 \leq f_d \leq 4095$
操作：	((FSR2) + z_s) → f_d
受影响的状态位：	无

机器码：

第一个字 （源）	`1110`	`1011`	`0zzz`	`zzzz`$_s$
第二个字 （目标）	`1111`	`ffff`	`ffff`	`ffff`$_d$

说明：	将源寄存器的内容移入目标寄存器 f_d。通过将第一个字中的 7 位立即数偏移量 z_s 与 FSR2 的值相加，来确定源寄存器的实际地址。第二个字中的 12 位立即数 f_d 指向目标寄存器的地址。两个地址均可以是 4096 字节的数据空间 （000h 到 FFFh）中的任何位置。 `MOVSF` 指令中的目标寄存器不能是 PCL、 TOSU、 TOSH 或 TOSL。 如果计算得到的源地址指向间接寻址寄存器，将返回 00h。
指令字数：	2
指令周期数：	2

Q 周期操作：

Q1	Q2	Q3	Q4
译码	确定源地址	确定源地址	读源寄存器
译码	空操作 非无效读取	空操作	写目标寄存器 f

示例： `MOVSF [05h], REG2`

执行指令前
- FSR2 = 80h
- 85h 单元的内容 = 33h
- REG2 = 11h

执行指令后
- FSR2 = 80h
- 85h 单元的内容 = 33h
- REG2 = 33h

MOVSS 变址寻址传送数据

语法： MOVSS [z_s], [z_d]

操作数： $0 \leq z_s \leq 127$
$0 \leq z_d \leq 127$

操作： ((FSR2) + z_s) → ((FSR2) + z_d)

受影响的状态位： 无

机器码：

第一个字 （源）	1110	1011	1zzz	$zzzz_s$
第二个字 （目标）	1111	xxxx	xzzz	$zzzz_d$

说明： 将源寄存器的内容送入目标寄存器。通过将 FSR2 中的值分别加上 7 位立即数偏移量 z_s 或 z_d 来确定源寄存器和目标寄存器的地址。两个寄存器都可以是 4096 字节数据存储空间 （000h 到 FFFh）中的任意单元。
MOVSS 指令中的目标寄存器不能是 PCL、 TOSU、 TOSH 或 TOSL。
如果计算得到的源地址指向间接寻址寄存器，将返回 00h。如果计算得到的目标地址指向间接寻址寄存器，指令将作为一条 NOP 指令执行。

指令字数： 2

指令周期数： 2

Q 周期操作：

Q1	Q2	Q3	Q4
译码	确定源地址	确定源地址	读源寄存器
译码	确定目标地址	确定目标地址	写目标寄存器

示例： `MOVSS [05h], [06h]`

执行指令前
FSR2 = 80h
85h 单元的内容 = 33h
86h 单元的内容 = 11h

执行指令后
FSR2 = 80h
85h 单元的内容 = 33h
86h 单元的内容 = 33h

PUSHL 将立即数保存在 FSR2，FSR2 减 1

语法： PUSHL k

操作数： $0 \leq k \leq 255$

操作： k → （FSR2），
FSR2 – 1 → FSR2

受影响的状态位：无

机器码：

1111	1010	kkkk	kkkk

说明： 8 位立即数 k 被写入由 FSR2 指定的数据存储单元。操作完后 FSR2 减 1。
此指令允许用户将值压入软件堆栈。

指令字数： 1

指令周期数： 1

Q 周期操作：

Q1	Q2	Q3	Q4
译码	读取 k	处理数据	写入目标寄存器

示例： `PUSHL 08h`

执行指令前
FSR2H:FSR2L = 01ECh
存储单元 （01ECh） = 00h

执行指令后
FSR2H:FSR2L = 01EBh
存储单元 （01ECh） = 08h

SUBFSR **FSR 减去立即数**

语法：	SUBFSR f, k
操作数：	0 ≤ k ≤ 63 f ∈ [0, 1, 2]
操作：	FSRf – k → FSRf
受影响的状态位：	无
机器码：	1110 | 1001 | ffkk | kkkk
说明：	用 f 指定的 FSR 的内容减去 6 位立即数 k。
指令字数：	1
指令周期数：	1

Q 周期操作：

Q1	Q2	Q3	Q4
译码	读 寄存器 f	处理数据	写入 目标寄存器

示例： `SUBFSR 2, 23h`

执行指令前
FSR2 = 03FFh

执行指令后
FSR2 = 03DCh

SUBULNK **FSR2 减去立即数并返回**

语法：	SUBULNK k
操作数：	0 ≤ k ≤ 63
操作：	FSR2 – k → FSR2 (TOS) → PC
受影响的状态位：	无
机器码：	1110 | 1001 | 11kk | kkkk
说明：	将 FSR2 的内容减去 6 位立即数 k，然后通过将 TOS 装入 PC，执行一条 RETURN 指令。 执行该指令需要两个指令周期，第二个指令周期执行一条 NOP 指令。 该指令可以被认为是 SUBFSR 指令的特例，其中 f = 3 （二进制 11）；它仅针对 FSR2 进行操作。
指令字数：	1
指令周期数：	2

Q 周期操作：

Q1	Q2	Q3	Q4
译码	读 寄存器 f	处理数据	写入 目标寄存器
空操作	空操作	空操作	空操作

示例： `SUBULNK 23h`

执行指令前
FSR2 = 03FFh
PC = 0100h

执行指令后
FSR2 = 03DCh
PC = (TOS)

26.2.3 立即数变址寻址模式中面向字节和位的指令

> **注：** 使能 PIC18 扩展指令集可能导致常规应用程序运行不正常或完全失败。

一旦使能扩展的指令集，除了可以使用 8 条新命令之外，还可以使用立即数变址寻址模式（**第 5.6.1 节“使用立即数偏移量进行变址寻址”**）。这将导致标准 PIC18 指令的地址解析方法有很大变化。

当禁用扩展的指令集时，被嵌入在操作码中的地址被视作立即数存储单元地址：可以是快速操作存储区中的单元（a = 0），或由 BSR 指定的 GPR 存储区中的单元（a = 1）。当使能扩展的指令集且 a = 0 时，地址为 5Fh 或以下的文件寄存器参数被解析为 FSR2 中的指针值的偏移量，而不是一个立即数地址。对于实际应用来说，这意味着所有使用快速操作 RAM 位作为参数的指令，即所有面向字节或位的指令，或者几乎半数的 PIC18 内核指令——在使能了扩展的指令集时操作都会有所不同。

当 FSR2 的内容为 00h 时，快速操作 RAM 的边界会被重新映射到它们的原始值。这对于编写向下兼容的代码很有用处。如果使用此技术，有必要在 C 程序调用汇编子程序时保存 FSR2 的值并在返回时将它恢复，这样做的目的是保护堆栈指针。用户还必须记住扩展指令集的语法要求（见**第 26.2.3.1 节“标准 PIC18 命令的扩展指令语法”**）。

虽然立即数变址寻址模式对于动态堆栈和指针控制很有用处，但是如果不小心误用了寄存器，也会非常麻烦。已经习惯使用 PIC18 编程的用户必须记住，在使能了扩展的指令集时，地址小于或等于 5Fh 的寄存器用于立即数变址寻址。

下面是在立即数变址寻址模式中，一些面向字节和位的指令的示例，通过示例可以看出指令如何受到影响。示例中的操作数条件适用于所有这一类的指令。

26.2.3.1 标准 PIC18 命令的扩展指令语法

当使能了扩展的指令集时，立即数偏移量 k 被用来替换标准的面向字节和位的命令中的文件寄存器参数 f。如前所述，只有在 f 小于或等于 5Fh 时才会发生这种情况。当使用偏移量时，该偏移量必须用方括号“[ ]”标出。因为在扩展的指令集中，括号中的数值被解析为变址地址或偏移量。省略括号，或在括号内使用大于 5Fh 的值会在 MPASM 汇编器中产生错误。

如果变址参数已被加上了括号，那么就不再需要指定快速操作 RAM 参数；此参数被假定为 0。这与标准操作（禁止扩展的指令集时）正好相反。在变址寻址模式中，声明快速操作 RAM 位也将在 MPASM 汇编器中产生错误。

目标参数 d 的操作和以前一样。

在 MPASM 汇编器的最新版本中，必须明确调用对扩展的指令集的语言支持。可以通过命令行选项 /y 或在源代码中加入 PE 伪指令进行调用。

26.2.4 使能扩展的指令集时的注意事项

需要注意的是并非所有用户都有必要使用扩展的指令集，尤其是那些不使用软件堆栈的用户。

此外，立即数变址寻址模式可能会给写入 PIC18 汇编器的常规应用程序带来问题。这是因为常规的指令会尝试寻址快速操作存储区中地址低于 5Fh 的寄存器。当使能了扩展的指令集时，这些地址被解析为相对于 FSR2 的立即数偏移量，所以应用程序会读或写错误的地址。

将应用程序移植到 PIC18F2455/2550/4455/4550 器件时，代码类型是非常重要的。在使用扩展的指令集时，用 C 语言编写的代码较长的重入应用程序会运行的很好，而大量使用快速操作存储区的常规应用程序不会获得任何益处。

初稿

ADDWF　将 W 与变址寻址单元的内容相加（立即数变址寻址模式）

语法：	ADDWF [k] {,d}
操作数：	0 ≤ k ≤ 95 d ∈ [0,1]
操作：	(W) + ((FSR2) + k) → dest
受影响的状态位：	N、OV、C、DC 和 Z
机器码：	0010 01d0 kkkk kkkk
说明：	将 W 的内容与由 FSR2 加上偏移量 k 指定的寄存器的内容相加。 如果 d 为 0，结果存储在 W 中。如果 d 为 1，结果存回寄存器 f（默认）。
指令字数：	1
指令周期数：	1

Q 周期操作：

Q1	Q2	Q3	Q4
译码	读取 k	处理数据	写入目标寄存器

示例：　`ADDWF [OFST] ,0`

执行指令前
- W = 17h
- OFST = 2Ch
- FSR2 = 0A00h
- 0A2Ch 单元的内容 = 20h

执行指令后
- W = 37h
- 0A2Ch 单元的内容 = 20h

BSF　将变址寻址单元相应位置 1（立即数变址寻址模式）

语法：	BSF [k], b
操作数：	0 ≤ f ≤ 95 0 ≤ b ≤ 7
操作：	1 → ((FSR2) + k)<b>
受影响的状态位：	无
机器码：	1000 bbb0 kkkk kkkk
说明：	将由 FSR2 加上偏移量 k 指定的寄存器中的位 b 置 1。
指令字数：	1
指令周期数：	1

Q 周期操作：

Q1	Q2	Q3	Q4
译码	读寄存器 f	处理数据	写入目标寄存器

示例：　`BSF [FLAG_OFST], 7`

执行指令前
- FLAG_OFST = 0Ah
- FSR2 = 0A00h
- 0A0Ah 单元的内容 = 55h

执行指令后
- 0A0Ah 单元的内容 = D5h

SETF　将变址寻址单元置全 1（立即数变址寻址模式）

语法：	SETF [k]
操作数：	0 ≤ k ≤ 95
操作：	FFh → ((FSR2) + k)
受影响的状态位：	无
机器码：	0110 1000 kkkk kkkk
说明：	将由 FSR2 加上偏移量 k 指定的寄存器的内容置为 FFh。
指令字数：	1
指令周期数：	1

Q 周期操作：

Q1	Q2	Q3	Q4
译码	读取 k	处理数据	写寄存器

示例：　`SETF [OFST]`

执行指令前
- OFST = 2Ch
- FSR2 = 0A00h
- 0A2Ch 单元的内容 = 00h

执行指令后
- 0A2Ch 单元的内容 = FFh

26.2.5 使用 MICROCHIP MPLAB® IDE 工具的注意事项

最新版本的Microchip软件工具，完全支持PIC18F2455/2550/4455/4550 系列器件的扩展指令集。包括 MPLAB C18 C 编译器、 MPASM 汇编语言和 MPLAB 集成开发环境（Integrated Development Environment ， IDE）。

在选择了用于软件开发的目标器件后， MPLAB IDE 将对该器件的默认配置进行自动设置。在禁用扩展的指令集和立即数变址寻址模式时，XINST 配置位的默认设置是 0。在编程过程中必须将 XINST 位置 1 才能正确使用扩展指令集开发应用程序。

要使用扩展的指令集开发软件，用户必须设置他们的语言工具以实现对扩展指令和变址寻址模式的支持。根据所使用的环境，可以通过以下几种方法：

- 开发环境中的菜单选项或对话框，允许用户配置项目的语言工具及其设置
- 命令行选项
- 源代码中的伪指令

这些选项在不同的编译器、汇编器和开发环境中将有所不同。建议用户在其开发系统所附带的文档中查询相应的信息。

27.0 开发支持

一系列硬件及软件开发工具对 PICmicro® 单片机提供支持：

- 集成开发环境
  - MPLAB® IDE 软件
- 汇编器 / 编译器 / 链接器
  - MPASM™ 汇编器
  - MPLAB C18 和 MPLAB C30 C 编译器
  - MPLINK™ 目标链接器 / MPLIB™ 目标库管理器
  - MPLAB ASM30 汇编器 / 链接器 / 库
- 模拟器
  - MPLAB SIM 软件模拟器
- 仿真器
  - MPLAB ICE 2000 在线仿真器
  - MPLAB ICE 4000 在线仿真器
- 在线调试器
  - MPLAB ICD 2
- 器件编程器
  - PICSTART® Plus 开发编程器
  - MPLAB PM3 器件编程器
  - PICkit™ 2 开发编程器
- 低成本演示和开发板及评估工具包

27.1 MPLAB 集成开发环境软件

MPLAB IDE 软件为 8/16 位单片机市场提供了前所未有的易于使用的软件开发平台。MPLAB IDE 是基于 Windows® 操作系统的应用软件，包括：

- 一个包含所有调试工具的图形界面
  - 模拟器
  - 编程器 （单独销售）
  - 仿真器 （单独销售）
  - 在线调试器 （单独销售）
- 具有彩色上下文代码显示的全功能编辑器
- 多项目管理器
- 内容可直接编辑的可定制式数据窗口
- 高级源代码调试
- 可视化器件初始化程序，便于进行寄存器的初始化
- 鼠标停留在变量上进行查看的功能
- 通过拖放把变量从源代码窗口拉到观察窗口
- 丰富的在线帮助
- 集成了可选的第三方工具，如 HI-TECH 软件 C 编译器和 IAR C 编译器

MPLAB IDE 可以让您：

- 编辑源文件 （汇编语言或 C 语言）
- 点击一次即可完成汇编 （或编译）并将代码下载到 PICmicro MCU 仿真器和模拟器工具中 （自动更新所有项目信息）
- 可使用如下各项进行调试：
  - 源文件 （汇编语言或 C 语言）
  - 混合汇编语言和 C 语言
  - 机器码

MPLAB IDE 在单个开发范例中支持使用多种调试工具，包括从成本效益高的模拟器到低成本的在线调试器，再到全功能的仿真器。这样缩短了用户升级到更加灵活而功能更强大的工具时的学习时间。

27.2 MPASM 汇编器

MPASM 汇编器是全功能通用宏汇编器，适用于所有的 PICmicro MCU。

MPASM 汇编器可生成用于 MPLINK 目标链接器的可重定位目标文件、Intel® 标准 HEX 文件、详细描述存储器使用状况和符号参考的 MAP 文件、包含源代码行及生成机器码的绝对 LST 文件以及用于调试的 COFF 文件。

MPASM 汇编器具有如下特征：

- 集成在 MPLAB IDE 项目中
- 用户定义的宏可简化汇编代码
- 对多用途源文件进行条件汇编
- 允许完全控制汇编过程的指令

27.3 MPLAB C18 和 MPLAB C30 C 编译器

MPLAB C18 和 MPLAB C30 代码开发系统是完全的 ANSI C 编译器，分别适用于 Microchip 的 PIC18 系列单片机及 dsPIC30F、dsPIC33 和 PIC24 系列数字信号控制器。这些编译器可提供其他编译器并不具备的强大的集成功能和出众的代码优化能力，且使用方便。

为便于源代码调试，编译器提供了针对 MPLAB IDE 调试器的优化符号信息。

27.4 MPLINK 目标链接器 / MPLIB 目标库管理器

MPLINK 目标链接器包含了由 MPASM 汇编器、MPLAB C18 C 编译器产生的可重定位目标。通过使用链接器脚本中的指令，它还可链接预编译库中的可重定位目标。

MPLIB目标库管理器管理预编译代码库文件的创建和修改。当从源文件调用库中的一段子程序时，只有包含此子程序的模块被链接到应用中。这样可使大型库在许多不同应用中被高效地利用。

目标链接器 / 库管理器具有如下特征：

- 高效地连接单个的库而不是许多小文件
- 通过将相关的模块组合在一起来增强代码的可维护性
- 只要列出、替换、删除和抽取模块，便可灵活地创建库

27.5 MPLAB ASM30 汇编器、链接器和库管理器

MPLAB ASM30 汇编器为 dsPIC30F 器件提供转换自符号汇编语言的可重定位机器码。 MPLAB C30 C 编译器使用该汇编器生成目标文件。汇编器产生可重定位目标文件之后，可将这些目标文件存档，或与其他可重定位目标文件和存档链接以生成可执行文件。该汇编器有如下显著特征：

- 支持整个 dsPIC30F 指令集
- 支持定点数据和浮点数据
- 命令行界面
- 丰富的指令集
- 灵活的宏语言
- MPLAB IDE 兼容性

27.6 MPLAB SIM 软件模拟器

MPLAB SIM 软件模拟器在指令级对 PICmicro MCU 和 dsPIC® DSC 进行模拟，使得用户可以在 PC 主机的环境下进行代码开发。对于任何给定的指令，用户均可对数据区进行检查或修改，并通过各种触发机制来产生激励。可以将各寄存器的情况记录在文件中，以便进行进一步地运行时分析。跟踪缓冲器和逻辑分析器的显示使模拟器还能记录和跟踪程序的执行、 I/O 的动作、大部分的外设及内部寄存器的状况。

MPLAB SIM 软件模拟器完全支持使用 MPLAB C18 和 MPLAB C30 C 编译器以及MPASM和MAPLAB ASM30 汇编器的符号调试。该软件模拟器可用于在硬件实验室环境外灵活地开发和调试代码，是一款完美且经济的软件开发工具。

27.7 MPLAB ICE 2000 高性能在线仿真器

MPLAB ICE 2000 在线仿真器旨在为产品开发工程师提供一整套用于 PICmicro 单片机的设计工具。 MPLAB ICE 2000 在线仿真器的软件控制由 MPLAB 集成开发环境平台提供，它允许在单一环境下进行编辑、编译、下载以及源代码调试。

MPLAB ICE 2000 是全功能仿真器系统，它具有增强的跟踪、触发和数据监控功能。处理器模块可插拔，使系统可轻松进行重新配置以适应各种不同处理器的仿真需要。MPLAB ICE 2000 在线仿真器的架构允许对其进行扩展以支持新的 PICmicro 单片机。

MPLAB ICE 2000 在线仿真器系统设计为一款实时仿真系统，该仿真系统具备通常只有昂贵的开发工具中才有的高级功能。选择 PC 平台和 Microsoft® Windows® 32 位操作系统可使这些功能在一个简单而统一的应用中得到很好的利用。

27.8 MPLAB ICE 4000 高性能在线仿真器

MPLAB ICE 4000 在线仿真器旨在为产品开发工程师提供一整套用于高端 PICmicro MCU 和 dsPIC DSC 的设计工具。 MPLAB ICE 4000 在线仿真器的软件控制由 MPLAB 集成开发环境平台提供，它允许在单一环境下进行编辑、编译、下载以及源代码调试。

MPLAB ICE 4000 是高级的仿真系统，除具备 MPLAB ICE 2000 的所有功能外，它还增加了适用于 dsPIC30F 和 PIC18XXXX 器件的仿真存储容量以及高速性能。该仿真器的先进特性包括复杂触发和定时功能及高达 2 Mb 的仿真存储容量。

MPLAB ICE 4000 在线仿真系统设计为一款实时仿真系统，该仿真系统具备通常只有在更加昂贵的开发工具中才有的高级功能。选择 PC 平台和 Microsoft Windows 32 位操作系统可使这些功能在一个简单而统一的应用程序中得以很好的利用。

27.9 MPLAB ICD 2 在线调试器

Microchip 的在线调试器 MPLAB ICD 2 是一款功能强大而成本低廉的运行时开发工具，通过 RS-232 或高速 USB 接口与 PC 主机相连。该工具基于闪存 PICmicro MCU，可用于开发本系列及其他 PICmicro MCU 和 dsPIC DSC。 MPLAB ICD 2 使用了闪存器件中内建的在线调试功能。该功能结合 Microchip 的在线串行编程（In-Circuit Serial Programming™， ICSP™）协议，可在 MPLAB 集成开发环境的图形用户界面上提供成本效益很高的在线闪存调试。这使设计人员可通过设置断点、单步运行以及对变量、CPU 状态以及外设寄存器进行监视的方法实现源代码的开发和调试。其全速运行特性可对硬件和应用进行实时测试。MPLAB ICD 2 还可用作某些 PICmicro 器件的开发编程器。

27.10 MPLAB PM3 器件编程器

MPLAB PM3 器件编程器是一款通用的、符合 CE 规范的器件编程器，其可编程电压设置在 V_{DDMIN} 和 V_{DDMAX} 之间时可靠性最高。它有一个用来显示菜单和错误信息的大 LCD 显示器（128 x 64），以及一个支持各种封装类型的可拆卸模块化插槽装置。编程器标准配置中带有一根 ICSP™ 电缆。在单机模式下， MPLAB PM3 器件编程器不必与 PC 相连即可对 PICmicro 器件进行读取、验证和编程。在该模式下它还可设置代码保护。MPLAB PM3 通过 RS-232 或 USB 电缆连接到 PC 主机上。MPLAB PM3 具备高速通信能力以及优化算法，可对存储器很大的器件进行快速编程，它还采用 SD/MMC 卡用作文件存储及数据安全应用。

27.11 PICSTART Plus 开发编程器

PICSTART Plus开发编程器是一款易于使用而成本低廉的原型编程器。它通过 COM （RS-232）端口与 PC 相连。 MPLAB 集成开发环境软件使得该编程器的使用简便、高效。PICSTART Plus 开发编程器支持采用 DIP 封装的大部分 PICmicro 器件，其引脚数最多可达 40 个。引脚数更多的器件，如 PIC16C92X 和 PIC17C76X，可通过连接一个转接插槽来获得支持。PICSTART Plus 开发编程器符合 CE 规范。

27.12 PICkit 2 开发编程器

PICkit™ 2 开发编程器是一个低成本编程器，通过其易于使用的接口可对众多 Microchip 的低档、中档和 PIC18F 系列闪存单片机进行编程。PICkit 2 入门工具包中包含一个有实验布线区的开发板、十二堂系列课程、软件和 HI-TECH 的 PICC Lite C 编译器，有助于用户快速掌握 PIC® 单片机的使用。这一工具包为使用 Microchip 功能强大的中档闪存系列单片机进行编程、评估和应用开发，提供了所需的一切。

27.13 演示、开发和评估板

有许多演示、开发和评估板可用于各种 PICmicro MCU 和 dsPIC DSC，实现对全功能系统的快速应用开发。大多数的演示、开发和评估板都有实验布线区，供用户添加定制电路；还有应用固件和源代码，用于测试和修改。

这些板支持多种功能部件，包括 LED、温度传感器、开关、扬声器、 RS-232 接口、 LCD 显示器、电位计和附加 EEPROM 存储器。

演示和开发板可用于教学环境，在实验布线区设计定制电路，从而掌握各种单片机应用。

除了 PICDEM™ 和 dsPICDEM™ 演示 / 开发板系列电路外， Microchip 还有一系列评估工具包和演示软件，适用于模拟滤波器设计、 KEELOQ® 数据安全产品 IC、CAN、 IrDA®、 PowerSmart® 电池管理、 SEEVAL® 评估系统、 Σ–Δ ADC、流速传感器，等等。

有关演示、开发和评估工具包的完整列表，请查阅 Microchip 公司网页 （www.microchip.com）以及最新的 *“Product Selector Guide（产品选型指南）”*（DS00148）。



注：

注：

28.0 电气规范

绝对最大值（†）

偏置电压下的环境温度 .... -40°C 至 +85°C
储存温度 .... -65°C 至 +150°C
任一引脚（除了 V_{DD}、$\overline{MCLR}$ 和 RA4 之外）相对于 V_{SS} 的电压 .... -0.3V 至（V_{DD} + 0.3V）
V_{DD} 相对于 V_{SS} 的电压 .... -0.3V 至 +7.5V
$\overline{MCLR}$ 引脚相对于 V_{SS} 的电压（**注 2**）........ 0V 至 +13.25V
总功耗（**注 1**）........ 1.0W
V_{SS} 引脚最大输出电流 .... 300 mA
V_{DD} 引脚最大输入电流 .... 250 mA
输入钳位电流 I_{IK}（$V_I < 0$ 或 $V_I > V_{DD}$）........ ± 20 mA
输出钳位电流 I_{OK}（$V_O < 0$ 或 $V_O > V_{DD}$）........ ± 20 mA
任一 I/O 引脚的最大灌电流 .... 25 mA
任一 I/O 引脚的最大拉电流 .... 25 mA
所有端口的最大灌电流 .... 200 mA
所有端口的最大拉电流 .... 200 mA

注 **1**：功耗按如下公式计算：
$Pdis = V_{DD} \times \{I_{DD} - \sum I_{OH}\} + \sum \{(V_{DD} - V_{OH}) \times I_{OH}\} + \sum (V_{OL} \times I_{OL})$

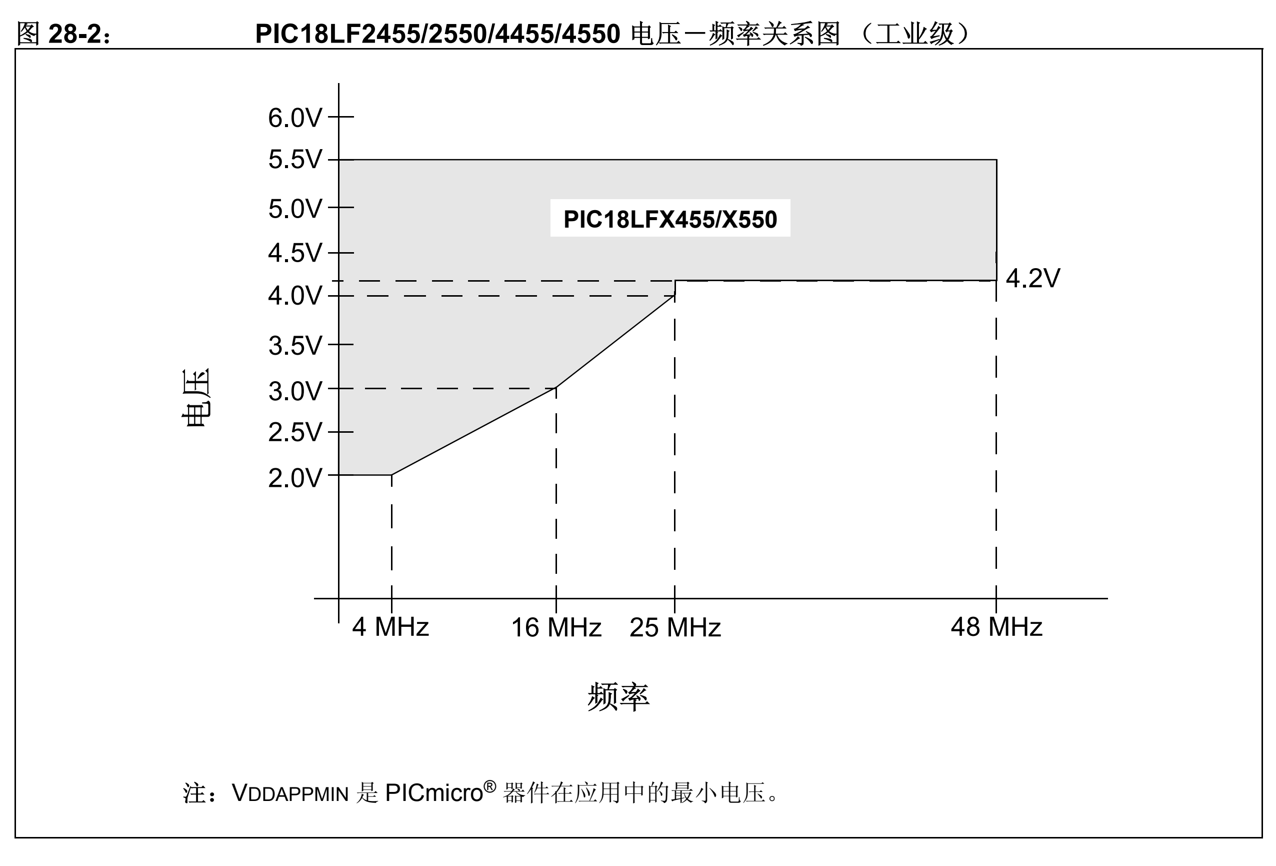
2：如果 $\overline{MCLR}$/VPP/RE3 引脚上的尖峰电压低于 V_{SS}，感应电流大于 80mA，可能会引起器件锁死。因此当 $\overline{MCLR}$/VPP/RE3 引脚驱动为低电平时，应该串联一个 50 － 100Ω 的电阻，而不是直接把该引脚连接到 V_{SS}。

† 注意： 如果器件工作条件超过上述“绝对最大值”，可能会对器件造成永久性损坏。上述值仅为运行条件极大值，我们不建议器件在该规范规定的范围以外运行。器件长时间工作在最大值条件下，其稳定性会受到影响。

图 28-1： **PIC18F2455/2550/4455/4550** 电压－频率关系图 （工业级）

6.0V
5.5V
5.0V
4.5V
4.0V
3.5V
3.0V
2.5V
2.0V
电压
PIC18FX455/X550
4.2V
48 MHz
频率

图 28-2： **PIC18LF2455/2550/4455/4550** 电压－频率关系图 （工业级）



注：$V_{DDAPPMIN}$ 是 PICmicro® 器件在应用中的最小电压。

初稿

28.1 直流规范： 供电电压 PIC18F2455/2550/4455/4550 （工业级） PIC18LF2455/2550/4455/4550 （工业级）

PIC18LF2455/2550/4455/4550 （工业级）			标准工作条件 （除非另外声明） 工作温度 -40°C ≤ TA ≤ +85°C （工业级）				
PIC18F2455/2550/4455/4550 （工业级）			标准工作条件 （除非另外声明） 工作温度 -40°C ≤ TA ≤ +85°C （工业级）				
参数编号	符号	特性	最小值	典型值	最大值	单位	条件
D001	VDD	供电电压	2.0	—	5.5	V	EC、 HS、 XT 和内部振荡器模式
			3.0	—	5.5	V	HSPLL、 XTPLL、 ECPIO 和 ECPLL 模式
D002	VDR	**RAM 数据保存电压** (1)	1.5	—	—	V	
D003	VPOR	**VDD 起动电压** 确保内部上电复位信号	—	—	0.7	V	详情请参见第 **4.3** 节 “上电复位 （**POR**）”。
D004	SVDD	**VDD 上升率** 确保内部上电复位信号	0.05	—	—	V/ms	详情请参见第 **4.3** 节 “上电复位 （**POR**）”。
D005	VBOR	欠压复位电压					
		BORV1:BORV0 = 11	2.00	2.05	2.16	V	
		BORV1:BORV0 = 10	2.65	2.79	2.93	V	
		BORV1:BORV0 = 01	4.11	4.33	4.55	V	
		BORV1:BORV0 = 00	4.36	4.59	4.82	V	

图注： 阴影行是为了增强表的可读性。

注 1： 该电压是在休眠模式或器件复位状态下，在不丢失 RAM 数据前提下的最小 VDD。

28.2 直流规范： 掉电电流和供电电流 PIC18F2455/2550/4455/4550 （工业级） PIC18LF2455/2550/4455/4550 （工业级）

PIC18LF2455/2550/4455/4550 （工业级）		标准工作条件 （除非另外声明） 工作温度 -40°C ≤ TA ≤ +85°C （工业级）				
PIC18F2455/2550/4455/4550 （工业级）		标准工作条件 （除非另外声明） 工作温度 -40°C ≤ TA ≤ +85°C （工业级）				
参数编号	**器件**	**典型值**	**最大值**	**单位**	**条件**	
	掉电电流 （IPD）(1)					
	PIC18LFX455/X550	0.1	0.95	μA	-40°C	VDD = 2.0V，（休眠模式）
		0.1	1.0	μA	+25°C	
		0.2	5	μA	+85°C	
	PIC18LFX455/X550	0.1	1.4	μA	-40°C	VDD = 3.0V，（休眠模式）
		0.1	2	μA	+25°C	
		0.3	8	μA	+85°C	
	所有器件	0.1	1.9	μA	-40°C	VDD = 5.0V，（休眠模式）
		0.1	2.0	μA	+25°C	
		0.4	15	μA	+85°C	

图注： 阴影行是为了增强表的可读性。

注 1： 休眠模式下的掉电电流不是由振荡器类型决定的。掉电电流是在器件休眠时，所有 I/O 引脚处于高阻态并且连接到 VDD 或者 VSS，以及禁止所有会带来新增电流的功能部件 （比如 WDT、 Timer1 振荡器和 BOR 等）时测得的。

2： 供电电流主要是由工作电压、频率和模式决定的。其他因素，如 I/O 引脚负载和开关频率、振荡器类型和电路、内部代码执行模式和温度也会影响电流消耗。
在正常工作模式下，所有 IDD 测量的测试条件为：
OSC1 = 外部方波，满幅；所有 I/O 引脚均为三态，拉至 VDD 或 VSS ；
MCLR=VDD ；根据具体应用使能或禁止 WDT。

3： 标准低成本 32 kHz 晶振的工作温度范围是从 -10°C 到 +70°C。扩展级温度的晶振成本要高很多。

4： BOR 和 HLVD 使能内部带隙参考源。当这两个模块同时被使能时，电流消耗将少于两个规范值之和。

28.2 直流规范： 掉电电流和供电电流 PIC18F2455/2550/4455/4550 （工业级） PIC18LF2455/2550/4455/4550 （工业级）（续）

PIC18LF2455/2550/4455/4550 （工业级）	标准工作条件 （除非另外声明） 工作温度 -40°C ≤ TA ≤ +85°C （工业级）
PIC18F2455/2550/4455/4550 （工业级）	标准工作条件 （除非另外声明） 工作温度 -40°C ≤ TA ≤ +85°C （工业级）

参数编号	器件	典型值	最大值	单位	条件		
	供电电流 （IDD）(2)						
	PIC18LFX455/X550	15	32	µA	-40°C	VDD = 2.0V	Fosc = 31 kHz （**RC_RUN** 模式， INTRC 时钟源）
		15	30	µA	+25°C		
		15	29	µA	+85°C		
	PIC18LFX455/X550	40	63	µA	-40°C	VDD = 3.0V	
		35	60	µA	+25°C		
		30	57	µA	+85°C		
	所有器件	105	168	µA	-40°C	VDD = 5.0V	
		90	160	µA	+25°C		
		80	152	µA	+85°C		
	PIC18LFX455/X550	0.33	1	mA	-40°C	VDD = 2.0V	Fosc = 1 MHz （**RC_RUN** 模式， INTOSC 时钟源）
		0.33	1	mA	+25°C		
		0.33	1	mA	+85°C		
	PIC18LFX455/X550	0.6	1.3	mA	-40°C	VDD = 3.0V	
		0.6	1.2	mA	+25°C		
		0.6	1.1	mA	+85°C		
	所有器件	1.1	2.3	mA	-40°C	VDD = 5.0V	
		1.1	2.2	mA	+25°C		
		1.0	2.1	mA	+85°C		
	PIC18LFX455/X550	0.8	2.1	µA	-40°C	VDD = 2.0V	Fosc = 4 MHz （**RC_RUN** 模式， INTOSC 时钟源）
		0.8	2.0	µA	+25°C		
		0.8	1.9	µA	+85°C		
	PIC18LFX455/X550	1.3	3.0	mA	-40°C	VDD = 3.0V	
		1.3	3.0	mA	+25°C		
		1.3	3.0	mA	+85°C		
	所有器件	2.5	5.3	mA	-40°C	VDD = 5.0V	
		2.5	5.0	mA	+25°C		
		2.5	4.8	mA	+85°C		

图注： 阴影行是为了增强表的可读性。

注 **1：** 休眠模式下的掉电电流不是由振荡器类型决定的。掉电电流是在器件休眠时，所有 I/O 引脚处于高阻态并且连接到 VDD 或者 VSS，以及禁止所有会带来新增电流的功能部件 （比如 WDT、 Timer1 振荡器和 BOR 等）时测得的。

2： 供电电流主要是由工作电压、频率和模式决定的。其他因素，如 I/O 引脚负载和开关频率、振荡器类型和电路、内部代码执行模式和温度也会影响电流消耗。
在正常工作模式下，所有 IDD 测量的测试条件为：
OSC1 = 外部方波，满幅；所有 I/O 引脚均为三态，拉至 VDD 或 VSS ；
MCLR=VDD ；根据具体应用使能或禁止 WDT。

3： 标准低成本 32 kHz 晶振的工作温度范围是从 -10°C 到 +70°C。扩展级温度的晶振成本要高很多。

4： BOR 和 HLVD 使能内部带隙参考源。当这两个模块同时被使能时，电流消耗将少于两个规范值之和。

 初稿

28.2 直流规范： 掉电电流和供电电流 PIC18F2455/2550/4455/4550 （工业级） PIC18LF2455/2550/4455/4550 （工业级） （续）

PIC18LF2455/2550/4455/4550 （工业级）		标准工作条件 （除非另外声明） 工作温度 -40°C ≤ TA ≤ +85°C （工业级）					
PIC18F2455/2550/4455/4550 （工业级）		标准工作条件 （除非另外声明） 工作温度 -40°C ≤ TA ≤ +85°C （工业级）					
参数编号	器件	典型值	最大值	单位	条件		
	供电电流 （IDD）(2)						
	PIC18LFX455/X550	2.9	8	μA	-40°C	VDD = 2.0V	FOSC = 31 kHz （**RC_IDLE** 模式，INTRC 时钟源）
		3.1	8	μA	+25°C		
		3.6	11	μA	+85°C		
	PIC18LFX455/X550	4.5	11	μA	-40°C	VDD = 3.0V	
		4.8	11	μA	+25°C		
		5.8	15	μA	+85°C		
	所有器件	9.2	16	μA	-40°C	VDD = 5.0V	
		9.8	16	μA	+25°C		
		11.4	36	μA	+85°C		
	PIC18LFX455/X550	165	350	μA	-40°C	VDD = 2.0V	FOSC = 1 MHz （**RC_IDLE** 模式，INTOSC 时钟源）
		175	350	μA	+25°C		
		190	350	μA	+85°C		
	PIC18LFX455/X550	250	500	μA	-40°C	VDD = 3.0V	
		270	500	μA	+25°C		
		290	500	μA	+85°C		
	所有器件	0.50	1	mA	-40°C	VDD = 5.0V	
		0.52	1	mA	+25°C		
		0.55	1	mA	+85°C		
	PIC18LFX455/X550	340	500	μA	-40°C	VDD = 2.0V	FOSC = 4 MHz （**RC_IDLE** 模式，INTOSC 时钟源）
		350	500	μA	+25°C		
		360	500	μA	+85°C		
	PIC18LFX455/X550	520	900	μA	-40°C	VDD = 3.0V	
		540	900	μA	+25°C		
		580	900	μA	+85°C		
	所有器件	1.0	1.6	mA	-40°C	VDD = 5.0V	
		1.1	1.5	mA	+25°C		
		1.1	1.4	mA	+85°C		

图注： 阴影行是为了增强表的可读性。

注 1： 休眠模式下的掉电电流不是由振荡器类型决定的。掉电电流是在器件休眠时，所有 I/O 引脚处于高阻态并且连接到 VDD 或者 VSS，以及禁止所有会带来新增电流的功能部件 （比如 WDT、 Timer1 振荡器和 BOR 等）时测得的。

2： 供电电流主要是由工作电压、频率和模式决定的。其他因素，如 I/O 引脚负载和开关频率、振荡器类型和电路、内部代码执行模式和温度也会影响电流消耗。
在正常工作模式下，所有 IDD 测量的测试条件为：
OSC1 = 外部方波，满幅；所有 I/O 引脚均为三态，拉至 VDD 或 VSS ；
MCLR=VDD ；根据具体应用使能或禁止 WDT。

3： 标准低成本 32 kHz 晶振的工作温度范围是从 -10°C 到 +70°C。扩展级温度的晶振成本要高很多。

4： BOR 和 HLVD 使能内部带隙参考源。当这两个模块同时被使能时，电流消耗将少于两个规范值之和。

28.2 直流规范： 掉电电流和供电电流 PIC18F2455/2550/4455/4550 （工业级） PIC18LF2455/2550/4455/4550 （工业级） （续）

PIC18LF2455/2550/4455/4550 （工业级）		标准工作条件 （除非另外声明） 工作温度 -40°C ≤ T_A ≤ +85°C （工业级）					
PIC18F2455/2550/4455/4550 （工业级）		标准工作条件 （除非另外声明） 工作温度 -40°C ≤ T_A ≤ +85°C （工业级）					
参数编号	器件	典型值	最大值	单位	条件		
	供电电流 （I_{DD}）(2)						
	PIC18LFX455/X550	250	500	μA	-40°C	V_{DD} = 2.0V	F_{OSC} = 1 MHz （**PRI_RUN**， EC 振荡器）
		250	500	μA	+25°C		
		250	500	μA	+85°C		
	PIC18LFX455/X550	550	650	μA	-40°C	V_{DD} = 3.0V	
		480	650	μA	+25°C		
		460	650	μA	+85°C		
	所有器件	1.2	1.6	mA	-40°C	V_{DD} = 5.0V	
		1.1	1.5	mA	+25°C		
		1.0	1.4	mA	+85°C		
	PIC18LFX455/X550	0.74	2.0	mA	-40°C	V_{DD} = 2.0V	F_{OSC} = 4 MHz （**PRI_RUN**， EC 振荡器）
		0.74	2.0	mA	+25°C		
		0.74	2.0	mA	+85°C		
	PIC18LFX455/X550	1.3	3.0	mA	-40°C	V_{DD} = 3.0V	
		1.3	3.0	mA	+25°C		
		1.3	3.0	mA	+85°C		
	所有器件	2.7	6.0	mA	-40°C	V_{DD} = 5.0V	
		2.6	6.0	mA	+25°C		
		2.5	6.0	mA	+85°C		
	所有器件	15	35	mA	-40°C	V_{DD} = 4.2V	F_{OSC} = 40 MHz （**PRI_RUN**， EC 振荡器）
		16	35	mA	+25°C		
		16	35	mA	+85°C		
	所有器件	21	40	mA	-40°C	V_{DD} = 5.0V	
		21	40	mA	+25°C		
		21	40	mA	+85°C		
	所有器件	20	40	mA	-40°C	V_{DD} = 4.2V	F_{OSC} = 48 MHz （**PRI_RUN**， EC 振荡器）
		20	40	mA	+25°C		
		20	40	mA	+85°C		
	所有器件	25	50	mA	-40°C	V_{DD} = 5.0V	
		25	50	mA	+25°C		
		25	50	mA	+85°C		

图注： 阴影行是为了增强表的可读性。

注 1： 休眠模式下的掉电电流不是由振荡器类型决定的。掉电电流是在器件休眠时，所有 I/O 引脚处于高阻态并且连接到 V_{DD} 或者 V_{SS}，以及禁止所有会带来新增电流的功能部件 （比如 WDT、 Timer1 振荡器和 BOR 等）时测得的。

2： 供电电流主要是由工作电压、频率和模式决定的。其他因素，如 I/O 引脚负载和开关频率、振荡器类型和电路、内部代码执行模式和温度也会影响电流消耗。
在正常工作模式下，所有 I_{DD} 测量的测试条件为：
OSC1 = 外部方波，满幅；所有 I/O 引脚均为三态，拉至 V_{DD} 或 V_{SS} ；
MCLR=V_{DD} ；根据具体应用使能或禁止 WDT。

3： 标准低成本 32 kHz 晶振的工作温度范围是从 -10°C 到 +70°C。扩展级温度的晶振成本要高很多。

4： BOR 和 HLVD 使能内部带隙参考源。当这两个模块同时被使能时，电流消耗将少于两个规范值之和。

28.2 直流规范： 掉电电流和供电电流 PIC18F2455/2550/4455/4550 （工业级） PIC18LF2455/2550/4455/4550 （工业级） （续）

PIC18LF2455/2550/4455/4550 （工业级）		标准工作条件 （除非另外声明） 工作温度 -40°C ≤ TA ≤ +85°C （工业级）					
PIC18F2455/2550/4455/4550 （工业级）		标准工作条件 （除非另外声明） 工作温度 -40°C ≤ TA ≤ +85°C （工业级）					
参数编号	器件	典型值	最大值	单位	条件		
	供电电流 （IDD）(2)						
	PIC18LFX455/X550	65	130	μA	-40°C	VDD = 2.0V	Fosc = 1 MHz （PRI_IDLE 模式，EC 振荡器）
		65	120	μA	+25°C		
		70	115	μA	+85°C		
	PIC18LFX455/X550	120	270	μA	-40°C	VDD = 3.0V	
		120	250	μA	+25°C		
		130	240	μA	+85°C		
	所有器件	230	480	μA	-40°C	VDD = 5.0V	
		240	450	μA	+25°C		
		250	430	μA	+85°C		
	PIC18LFX455/X550	255	475	μA	-40°C	VDD = 2.0V	Fosc = 4 MHz （PRI_IDLE 模式，EC 振荡器）
		260	450	μA	+25°C		
		270	430	μA	+85°C		
	PIC18LFX455/X550	420	900	μA	-40°C	VDD = 3.0V	
		430	850	μA	+25°C		
		450	810	μA	+85°C		
	所有器件	0.9	1.5	mA	-40°C	VDD = 5.0V	
		0.9	1.4	mA	+25°C		
		0.9	1.3	mA	+85°C		
	所有器件	6.0	16	mA	-40°C	VDD = 4.2V	Fosc = 40 MHz （PRI_IDLE 模式，EC 振荡器）
		6.2	16	mA	+25°C		
		6.6	16	mA	+85°C		
	所有器件	8.1	18	mA	-40°C	VDD = 5.0V	
		8.3	18	mA	+25°C		
		9.0	18	mA	+85°C		
	所有器件	8.0	18	mA	-40°C	VDD = 4.2V	Fosc = 48 MHz （PRI_IDLE 模式，EC 振荡器）
		8.1	18	mA	+25°C		
		8.2	18	mA	+85°C		
	所有器件	9.8	21	mA	-40°C	VDD = 5.0V	
		10.0	21	mA	+25°C		
		10.5	21	mA	+85°C		

图注： 阴影行是为了增强表的可读性。

注 1： 休眠模式下的掉电电流不是由振荡器类型决定的。掉电电流是在器件休眠时，所有 I/O 引脚处于高阻态并且连接到 VDD 或者 VSS，以及禁止所有会带来新增电流的功能部件 （比如 WDT、 Timer1 振荡器和 BOR 等）时测得的。

2： 供电电流主要是由工作电压、频率和模式决定的。其他因素，如 I/O 引脚负载和开关频率、振荡器类型和电路、内部代码执行模式和温度也会影响电流消耗。
在正常工作模式下，所有 IDD 测量的测试条件为：
OSC1 = 外部方波，满幅；所有 I/O 引脚均为三态，拉至 VDD 或 VSS ；
MCLR=VDD ；根据具体应用使能或禁止 WDT。

3： 标准低成本 32 kHz 晶振的工作温度范围是从 -10°C 到 +70°C。扩展级温度的晶振成本要高很多。

4： BOR 和 HLVD 使能内部带隙参考源。当这两个模块同时被使能时，电流消耗将少于两个规范值之和。

28.2 直流规范： 掉电电流和供电电流 PIC18F2455/2550/4455/4550 （工业级） PIC18LF2455/2550/4455/4550 （工业级） （续）

PIC18LF2455/2550/4455/4550 （工业级）	标准工作条件 （除非另外声明） 工作温度 -40°C ≤ TA ≤ +85°C （工业级）
PIC18F2455/2550/4455/4550 （工业级）	标准工作条件 （除非另外声明） 工作温度 -40°C ≤ TA ≤ +85°C （工业级）

参数编号	器件	典型值	最大值	单位	条件		
	供电电流 （IDD）(2)						
	PIC18LFX455/X550	14	40	μA	-40°C	VDD = 2.0V	FOSC = 32 kHz(3)（**SEC_RUN** 模式，Timer1 作为时钟源）
		15	40	μA	+25°C		
		16	40	μA	+85°C		
	PIC18LFX455/X550	40	74	μA	-40°C	VDD = 3.0V	
		35	70	μA	+25°C		
		31	67	μA	+85°C		
	所有器件	99	150	μA	-40°C	VDD = 5.0V	
		81	150	μA	+25°C		
		75	150	μA	+85°C		
	PIC18LFX455/X550	2.5	12	μA	-40°C	VDD = 2.0V	FOSC = 32 kHz(3)（**SEC_IDLE** 模式，Timer1 作为时钟源）
		3.7	12	μA	+25°C		
		4.5	12	μA	+85°C		
	PIC18LFX455/X550	5.0	15	μA	-40°C	VDD = 3.0V	
		5.4	15	μA	+25°C		
		6.3	15	μA	+85°C		
	所有器件	8.5	25	μA	-40°C	VDD = 5.0V	
		9.0	25	μA	+25°C		
		10.5	36	μA	+85°C		

图注： 阴影行是为了增强表的可读性。

注 **1**： 休眠模式下的掉电电流不是由振荡器类型决定的。掉电电流是在器件休眠时，所有 I/O 引脚处于高阻态并且连接到 VDD 或者 VSS，以及禁止所有会带来新增电流的功能部件 （比如 WDT、 Timer1 振荡器和 BOR 等）时测得的。

2： 供电电流主要是由工作电压、频率和模式决定的。其他因素，如 I/O 引脚负载和开关频率、振荡器类型和电路、内部代码执行模式和温度也会影响电流消耗。
在正常工作模式下，所有 IDD 测量的测试条件为：
OSC1 = 外部方波，满幅；所有 I/O 引脚均为三态，拉至 VDD 或 VSS；
MCLR=VDD；根据具体应用使能或禁止 WDT。

3： 标准低成本 32 kHz 晶振的工作温度范围是从 -10°C 到 +70°C。扩展级温度的晶振成本要高很多。

4： BOR 和 HLVD 使能内部带隙参考源。当这两个模块同时被使能时，电流消耗将少于两个规范值之和。

28.2 直流规范： 掉电电流和供电电流 PIC18F2455/2550/4455/4550 （工业级） PIC18LF2455/2550/4455/4550 （工业级）（续）

PIC18LF2455/2550/4455/4550 （工业级）		标准工作条件 （除非另外声明） 工作温度 -40°C ≤ TA ≤ +85°C （工业级）					
PIC18F2455/2550/4455/4550 （工业级）		标准工作条件 （除非另外声明） 工作温度 -40°C ≤ TA ≤ +85°C （工业级）					
参数编号	器件	典型值	最大值	单位	条件		
D022 （ΔIWDT）	模块差分电流 （ΔIWDT， ΔIBOR， ΔILVD， ΔIOSCB， ΔIAD）						
	看门狗定时器	1.3	3.8	μA	-40°C	VDD = 2.0V	
		1.4	3.8	μA	+25°C		
		2.0	3.8	μA	+85°C		
		1.9	4.6	μA	-40°C	VDD = 3.0V	
		2.0	4.6	μA	+25°C		
		2.8	4.6	μA	+85°C		
		4.0	10	μA	-40°C	VDD = 5.0V	
		5.5	10	μA	+25°C		
		5.6	10	μA	+85°C		
D022A （ΔIBOR）	欠压复位 [4]	35	40	μA	-40°C 至 +85°C	VDD = 3.0V	
		40	45	μA	-40°C 至 +85°C	VDD = 5.0V	
		0	2	μA	-40°C 至 +85°C		休眠模式， BOREN1:BOREN0 = 10
D022B （ΔILVD）	高低压检测 [4]	22	38	μA	-40°C 至 +85°C	VDD = 2.0V	
		25	40	μA	-40°C 至 +85°C	VDD = 3.0V	
		29	45	μA	-40°C 至 +85°C	VDD = 5.0V	
D025 （ΔIOSCB）	**Timer1 振荡器**	2.1	4.5	μA	-40°C	VDD = 2.0V	Timer1 为 32 kHz[3]
		1.8	4.5	μA	+25°C		
		2.1	4.5	μA	+85°C		
		2.2	6.0	μA	-40°C	VDD = 3.0V	Timer1 为 32 kHz[3]
		2.6	6.0	μA	+25°C		
		2.9	6.0	μA	+85°C		
		3.0	8.0	μA	-40°C	VDD = 5.0V	Timer1 为 32 kHz[3]
		3.2	8.0	μA	+25°C		
		3.4	8.0	μA	+85°C		
D026 （ΔIAD）	**A/D 转换器**	1.0	2.0	μA	-40°C 至 +85°C	VDD = 2.0V	A/D 启动，但不进行转换
		1.0	2.0	μA	-40°C 至 +85°C	VDD = 3.0V	
		1.0	2.0	μA	-40°C 至 +85°C	VDD = 5.0V	

图注： 阴影行是为了增强表的可读性。

注 1： 休眠模式下的掉电电流不是由振荡器类型决定的。掉电电流是在器件休眠时，所有 I/O 引脚处于高阻态并且连接到 VDD 或者 VSS，以及禁止所有会带来新增电流的功能部件 （比如 WDT、 Timer1 振荡器和 BOR 等）时测得的。

2： 供电电流主要是由工作电压、频率和模式决定的。其他因素，如 I/O 引脚负载和开关频率、振荡器类型和电路、内部代码执行模式和温度也会影响电流消耗。
在正常工作模式下，所有 IDD 测量的测试条件为：
OSC1 = 外部方波，满幅；所有 I/O 引脚均为三态，拉至 VDD 或 VSS ；
MCLR=VDD ；根据具体应用使能或禁止 WDT。

3： 标准低成本 32 kHz 晶振的工作温度范围是从 -10°C 到 +70°C。扩展级温度的晶振成本要高很多。

4： BOR 和 HLVD 使能内部带隙参考源。当这两个模块同时被使能时，电流消耗将少于两个规范值之和。

28.2 直流规范：掉电电流和供电电流 PIC18F2455/2550/4455/4550 （工业级） PIC18LF2455/2550/4455/4550 （工业级） （续）

PIC18LF2455/2550/4455/4550 （工业级）		标准工作条件 （除非另外声明） 工作温度 -40°C ≤ T_A ≤ +85°C （工业级）					
PIC18F2455/2550/4455/4550 （工业级）		标准工作条件 （除非另外声明） 工作温度 -40°C ≤ T_A ≤ +85°C （工业级）					
参数编号	器件	典型值	最大值	单位	条件		
	USB 和相关模块差分电流 （ΔI_{USBX}，ΔI_{PLL}，ΔI_{UREG}）						
ΔI_{USBX}	内置收发器的 **USB** 模块	TBD	TBD	mA	+25°C	V_{DD} = 3.3V	
		TBD	TBD	mA	+25°C	V_{DD} = 5.0V	
ΔI_{PLL}	**96 MHz PLL** （振荡模块）	TBD	TBD	TBD	+25°C	V_{DD} = 3.3V	
		TBD	TBD	TBD	+25°C	V_{DD} = 5.0V	
ΔI_{UREG}	**USB** 内部稳压器	TBD	TBD	TBD	+25°C	V_{DD} = 5.0V	

图注： 阴影行是为了增强表的可读性。

注 1： 休眠模式下的掉电电流不是由振荡器类型决定的。掉电电流是在器件休眠时，所有 I/O 引脚处于高阻态并且连接到 V_{DD} 或者 V_{SS}，以及禁止所有会带来新增电流的功能部件 （比如 WDT、 Timer1 振荡器和 BOR 等）时测得的。

2： 供电电流主要是由工作电压、频率和模式决定的。其他因素，如 I/O 引脚负载和开关频率、振荡器类型和电路、内部代码执行模式和温度也会影响电流消耗。
在正常工作模式下，所有 I_{DD} 测量的测试条件为：
OSC1 = 外部方波，满幅；所有 I/O 引脚均为三态，拉至 V_{DD} 或 V_{SS}；
MCLR=V_{DD}；根据具体应用使能或禁止 WDT。

3： 标准低成本 32 kHz 晶振的工作温度范围是从 -10°C 到 +70°C。扩展级温度的晶振成本要高很多。

4： BOR 和 HLVD 使能内部带隙参考源。当这两个模块同时被使能时，电流消耗将少于两个规范值之和。

28.3 直流规范： PIC18F2455/2550/4455/4550 （工业级） PIC18LF2455/2550/4455/4550 （工业级）

直流规范			标准工作条件 （除非另外声明） 工作温度 -40°C ≤ TA ≤ +85°C （工业级）			
参数编号	符号	特性	最小值	最大值	单位	条件
	VIL	输入低电压				
		I/O 端口 （USB 模式下的 RC4/RC5 除外）：				
D030		带 TTL 缓冲器	VSS	0.15 VDD	V	VDD < 4.5V
D030A			—	0.8	V	4.5V ≤ VDD ≤ 5.5V
D031		带施密特触发缓冲器 RC3 和 RC4	VSS VSS	0.2 VDD 0.3 VDD	V V	
D032		$\overline{\text{MCLR}}$	VSS	0.2 VDD	V	
D032A		OSC1 和 T1OSI	VSS	0.3 VDD	V	XT、 HS 和 HSPLL 模式 [1]
D033		OSC1	VSS	0.2 VDD	V	EC 模式 [1]
	VILU	D+/D- 输入	—	0.8	V	VDD = 4.35V，禁止 USB[5]
	VIH	输入高电压				
		I/O 端口 （USB 模式下的 RC4/RC5 除外）：				
D040		带 TTL 缓冲器	0.25 VDD + 0.8V	VDD	V	VDD < 4.5V
D040A			2.0	VDD	V	4.5V ≤ VDD ≤ 5.5V
D041		带施密特触发缓冲器 RC3 和 RC4	0.8 VDD 0.7 VDD	VDD VDD	V V	
D042		$\overline{\text{MCLR}}$	0.8 VDD	VDD	V	
D042A		OSC1 和 T1OSI	0.7 VDD	VDD	V	XT、 HS 和 HSPLL 模式 [1]
D043		OSC1	0.8 VDD	VDD	V	EC 模式 [1]
	VIHU	D+/D- 输入	2.4	—	V	VDD = 4.35V，禁止 USB[5]
	IIL	输入泄漏电流 [2,3]				
D060		I/O 端口	—	±1	mA	VSS ≤ VPIN ≤ VDD，引脚处于高阻态
D061		$\overline{\text{MCLR}}$	—	±5	mA	Vss ≤ VPIN ≤ VDD
D063		OSC1	—	±5	mA	Vss ≤ VPIN ≤ VDD
	IPU	弱上拉电流				
D070	IPURB	PORTB 弱上拉电流	50	400	mA	VDD = 5V， VPIN = VSS

注 1： 在 RC 振荡器配置中， OSC1/CLKI 引脚为施密特触发器输入。在 RC 模式下，建议不要使用外部时钟驱动 PICmicro® 器件。

2： $\overline{\text{MCLR}}$ 引脚上的泄漏电流主要由施加在该引脚上的电平决定。规定电平为正常工作条件下的电平。在不同的输入电压下可测得更高的泄漏电流。

3： 负电流定义为自引脚流出的电流。

4： 参数仅为特征值，未经测试。

5： D+ 参数依据 USB2.0 规范取值。

 初稿

28.3 直流规范： PIC18F2455/2550/4455/4550 （工业级） PIC18LF2455/2550/4455/4550 （工业级） （续）

直流规范			标准工作条件 （除非另外声明） 工作温度 -40°C ≤ T_A ≤ +85°C （工业级）			
参数编号	符号	特性	最小值	最大值	单位	条件
	V_{OL}	输出低电压				
D080		I/O 端口 （USB 模式下的 RC4/RC5 除外）：	—	0.6	V	I_{OL} = 8.5 mA， V_{DD} = 4.5V， -40°C 至 +85°C
D083		OSC2/CLKO （EC 和 ECIO 模式）	—	0.6	V	I_{OL} = 1.6 mA， V_{DD} = 4.5V， -40°C 至 +85°C
	V_{OLU}	D+/D- 输出	—	0.3		V_{DD} = 4.35V， 禁止 USB[5]
	V_{OH}	输出高电压 [3]				
D090		I/O 端口 （USB 模式下的 RC4/RC5 除外）：	V_{DD} - 0.7	—	V	I_{OH} = -3.0 mA，V_{DD} = 4.5V， -40°C 至 +85°C
D092		OSC2/CLKO （EC、 ECIO 和 ECPIO 模式）	V_{DD} - 0.7	—	V	I_{OH} = -1.3 mA，V_{DD} = 4.5V， -40°C 至 +85°C
	V_{OHU}	D+/D- 输出	2.8	3.6	V	V_{DD} = 4.35V， 禁止 USB[5]
		输出引脚上的容性负载规范				
D100[4]	C_{OSC2}	OSC2 引脚	—	15	pF	当外部时钟用于驱动 OSC1 时，处于 XT 和 HS 模式
D101	C_{IO}	所有 I/O 引脚和 OSC2 （在 RC 模式下）	—	50	pF	满足交流时序规范
D102	C_B	SCL， SDA	—	400	pF	I^2C™ 规范

注 1： 在 RC 振荡器配置中， OSC1/CLKI 引脚为施密特触发器输入。在 RC 模式下，建议不要使用外部时钟驱动 PICmicro® 器件。

2： $\overline{MCLR}$ 引脚上的泄漏电流主要由施加在该引脚上的电平决定。规定电平为正常工作条件下的电平。在不同的输入电压下可测得更高的泄漏电流。

3： 负电流定义为自引脚流出的电流。

4： 参数仅为特征值，未经测试。

5： D+ 参数依据 USB2.0 规范取值。

表 **28-1**： 存储器编程要求

直流规范			标准工作条件 （除非另外声明） 工作温度 -40°C ≤ TA ≤ +85°C （工业级）				
参数编号	符号	特性	最小值	典型值 †	最大值	单位	条件
		内部程序存储器编程规范 (1)					
D110	VPP	$\overline{\text{MCLR}}$/VPP/RE3 引脚上的电压	9.00	—	13.25	V	（注 **3**）
D113	IDDP	编程期间的供电电流	—	—	10	mA	
		数据 EEPROM 存储器					
D120	ED	字节的耐擦写能力	100K	1M	—	E/W	-40°C 至 +85°C
D121	VDRW	用于读 / 写的 VDD	VMIN	—	5.5	V	使用 EECON 读 / 写 VMIN = 最小工作电压
D122	TDEW	擦 / 写周期时间	—	4	—	ms	
D123	TRETD	保存时间	40	—	—	年	假如没有违反其他规范
D124	TREF	在刷新之前的总擦写周期数 (2)	1M	10M	—	E/W	-40°C 至 +85°C
		闪存程序存储器					
D130	EP	耐擦写能力	10K	100K	—	E/W	-40°C 至 +85°C
D131	VPR	用于读入的 VDD	VMIN	—	5.5	V	VMIN = 最小工作电压
D132	VIE	用于块擦除的 VDD	4.5	—	5.5	V	使用 ICSP™ 端口
D132A	VIW	用于外部定时擦写的 VDD	4.5	—	5.5	V	使用 ICSP 端口
D132B	VPEW	用于自定时写入的 VDD	VMIN	—	5.5	V	VMIN = 最小工作电压
D133	TIE	ICSP 块擦除周期时间	—	4	—	ms	VDD > 4.5V
D133A	TIW	ICSP 擦写周期时间 （外部定时）	1	—	—	ms	VDD > 4.5V
D133A	TIW	自定时写周期时间	—	2	—	ms	
D134	TRETD	保存时间	40	100	—	年	假如没有违反其他规范

† 除非另外声明，“典型值”栏中的数据均为 5.0V， 25°C 下的值。这些参数仅作为设计参考，未经测试。

注 **1**： 这些编程规范适用于通过表写指令对片上程序存储器编程的情况。

2： 有关 EEPROM 耐擦写能力的更多细节，请参见第 **7.7** 节 **“使用数据 EEPROM”**。

3： 只有在单电源编程被禁止时才需要。

表 **28-2**： 比较器规范

工作条件：除非另外声明，否则均为 3.0V < V_{DD} < 5.5V，-40°C < T_A < +85°C							
参数编号	符号	特性	最小值	典型值	最大值	单位	备注
D300	V_{IOFF}	输入失调电压	—	±5.0	±10	mV	
D301	V_{ICM}	输入共模电压 *	0	—	V_{DD} - 1.5	V	
D302	CMRR	共模抑制比 *	55	—	—	dB	
300	T_{RESP}	响应时间 [1]*	—	150	400	ns	PIC18**F**XXXX
300A			—	150	600	ns	PIC18**LF**XXXX，V_{DD} = 2.0V
301	T_{MC2OV}	比较器模式改变到输出有效的时间 *	—	—	10	µs	

* 这些参数仅为特征值，未经测试。

注 **1**： 响应时间是在比较器的一个输入端电压为 (V_{DD} – 1.5)/2，而另一个输入端从 V_{SS} 跳变到 V_{DD} 时测得的。

表 **28-3**： 参考电压规范

工作条件：除非另外声明，否则均为 3.0V < V_{DD} < 5.5V，-40°C < T_A < +85°C							
参数编号	符号	特性	最小值	典型值	最大值	单位	备注
D310	V_{RES}	分辨率	V_{DD}/24	—	V_{DD}/32	LSb	
D311	V_{RAA}	绝对精度	— —	1/4 —	1 1/2	LSb LSb	低电压范围 （CVRR = 1） 高电压范围 （CVRR = 0）
D312	V_{RUR}	单位电阻值 （R） *	—	2k	—	Ω	
310	T_{SET}	稳定时间 [1]*	—	—	10	µs	

* 这些参数仅为特征值，未经测试。

注 **1**： 稳定时间是在 CVRR = 1 并且 CVR3:CVR0 从 0000 跳变到 1111 时测得的。

表 28-4： USB 模块规范

工作条件：除非另外声明，否则均为 -40°C < TA < +85°C。							
参数编号	符号	特性	最小值	典型值	最大值	单位	备注
D313	VUSB	USB 电压	3.0	—	3.6	V	为保证 USB 正常工作，总线上的电压也必须在该范围内。
D314	IIL	引脚输入泄漏电流	—	—	±1	μA	VSS ≤ VPAD ≤ VDD；引脚处于高阻态
D315	VILUSB	USB 缓冲器的输入低电压	—	—	0.8	V	适用于 VUSB 范围
D316	VIHUSB	USB 缓冲器的输入高电压	2.0	—	—	V	适用于 VUSB 范围
D317	VCRS	交叉电压	1.3		2.0	V	用于产生 pad_dp 和 pad_dm 交叉的电压范围
D318	VDIFS	差动输入的灵敏度	—	—	0.2	V	当满足 VCM 时， D+ 和 D- 之差必须大于该值
D319	VCM	差分共模范围	0.8		2.5	V	
D320	ZOUT	驱动器输出阻抗	28	—	44	Ω	
D321	VOL	输出低电压	0.0	—	0.3	V	通过 1.5 kΩ 负载连接到 3.6V
D322	VOH	输出高电压	2.8	—	3.6	V	通过 15 kΩ 负载接地

表 28-5： USB 内部稳压器规范

工作条件：除非另外声明，否则均为 -40°C < TA < +85°C。							
参数编号	符号	特性	最小值	典型值	最大值	单位	备注
D323	VUSBANA	稳压器输出电压 *	3.0	—	3.6	V	
D324	CUSB	外部滤波器电容值 *	220	—	—	nF	必须充分充电，以保证在最高负荷时压降很小。

* 这些参数仅为特征值，未经测试。尚未给这些规范分配参数编号。

图 28-3： 高 / 低压检测特性

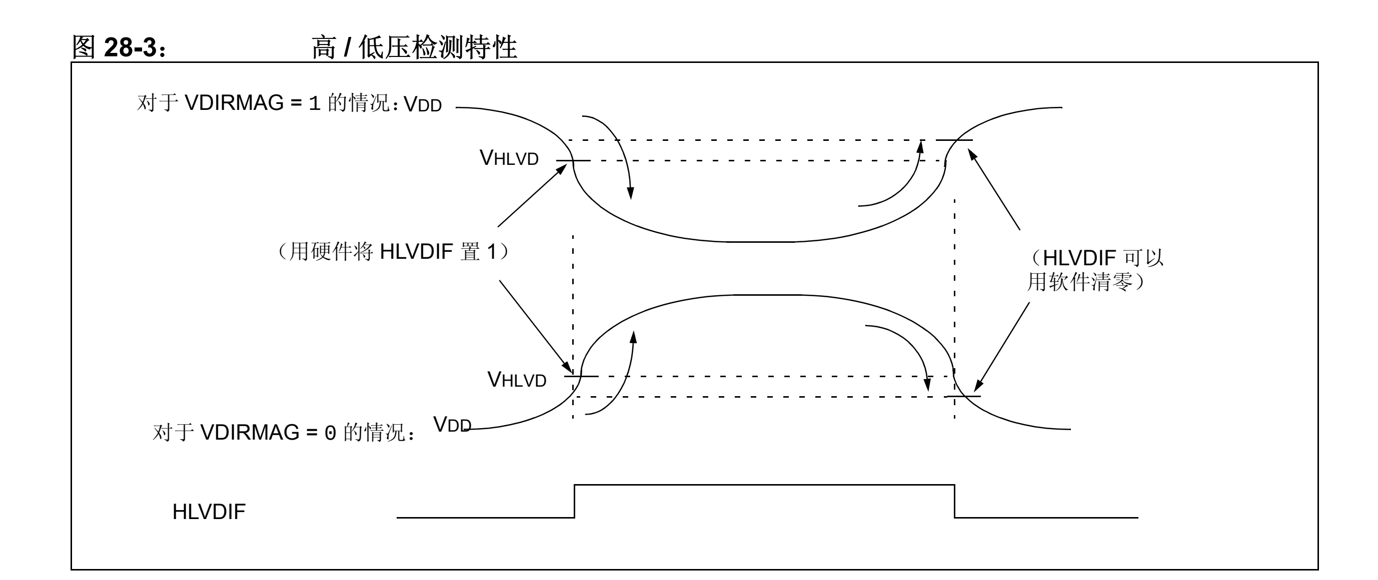


表 28-6： 高 / 低压检测特性

标准工作条件 （除非另外声明） 工作温度 -40°C ≤ TA ≤ +85°C （工业级）								
参数编号	符号	特性		最小值	典型值	最大值	单位	条件
D420		VDD 由高变为低时的 HLVD 电压	HLVDL = 0000	2.06	2.17	2.28	V	
			HLVDL = 0001	2.12	2.23	2.34	V	
			HLVDL = 0010	2.24	2.36	2.48	V	
			HLVDL = 0011	2.32	2.44	2.56	V	
			HLVDL = 0100	2.47	2.60	2.73	V	
			HLVDL = 0101	2.65	2.79	2.93	V	
			HLVDL = 0110	2.74	2.89	3.04	V	
			HLVDL = 0111	2.96	3.12	3.28	V	
			HLVDL = 1000	3.22	3.39	3.56	V	
			HLVDL = 1001	3.37	3.55	3.73	V	
			HLVDL = 1010	3.52	3.71	3.90	V	
			HLVDL = 1011	3.70	3.90	4.10	V	
			HLVDL = 1100	3.90	4.11	4.32	V	
			HLVDL = 1101	4.11	4.33	4.55	V	
			HLVDL = 1110	4.36	4.59	4.82	V	

28.4 交流（时序）规范

28.4.1 时序参数符号

可根据以下任一格式来创建时序参数符号：

1. TppS2ppS
2. TppS
3. TCC:ST （仅用于 I^2C 规范）
4. Ts （仅用于 I^2C 规范）

T			
F	频率	T	时间

小写字母（pp）和它们的含义：

pp			
ad	SPP 地址写	mc	$\overline{MCLR}$
cc	CCP1	osc	OSC1
ck	CLKO	rd	$\overline{RD}$
cs	$\overline{CS}$	rw	$\overline{RD}$ 或 $\overline{WR}$
da	SPP 数据写	sc	SCK
di	SDI	ss	$\overline{SS}$
do	SDO	t0	T0CKI
dt	数据输入	t1	T13CKI
io	I/O 端口	wr	$\overline{WR}$

大写字母和它们的含义：

S			
F	下降	P	周期
H	高	R	上升
I	无效（高阻态）	V	有效
L	低	Z	高阻态
仅用于 I^2C 模式			
AA	输出通道	High	高
BUF	总线空闲	Low	低

TCC：ST（仅用于 I^2C 规范）

CC			
HD	保持	SU	建立
ST			
DAT	保持数据输入	STO	停止条件
STA	启动条件		

28.4.2 时序条件

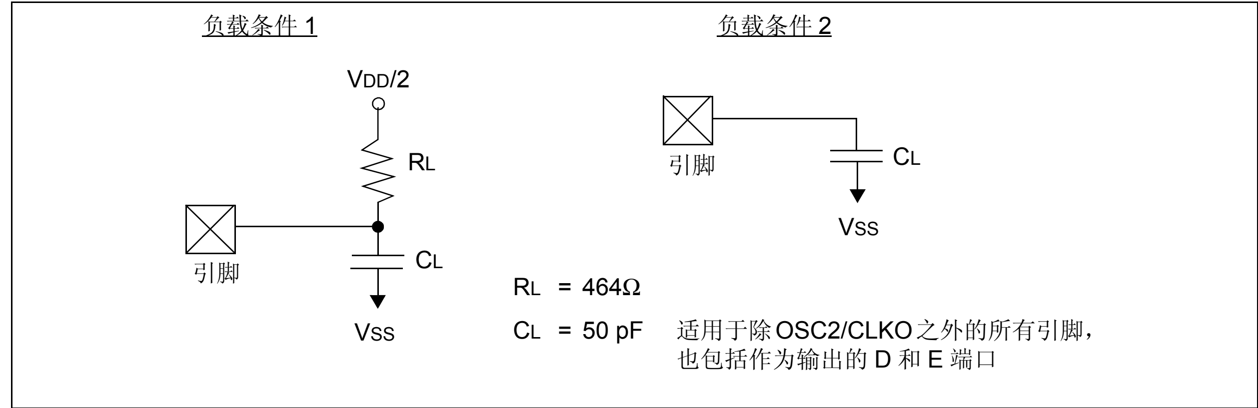
除非另外指明，表 28-7 中指定的温度和电压适用于所有时序规范。图 28-4 规定了时序规范的负载条件。

注： 由于篇幅所限，本章节中通称的“PIC18FXXXX”和“PIC18LFXXXX”特指（而且仅指代）PIC18F2455/2550/4455/4550 和 PIC18LF2455/2550/4455/4550 系列器件。

表 28-7： 温度和电压规范－交流

交流规范	**标准工作条件 （除非另外声明）** 工作温度 -40°C ≤ T_A ≤ +85°C （工业级） 直流规范第 **28.1** 节和第 **28.3** 节描述了工作电压 V_{DD} 的范围。 LF 器件仅在工业级温度下工作。

图 28-4： 器件时序规范的负载条件



28.4.3 时序图和规范

图 **28-5**： 外部时钟时序 （除 **PLL** 之外的所有模式）

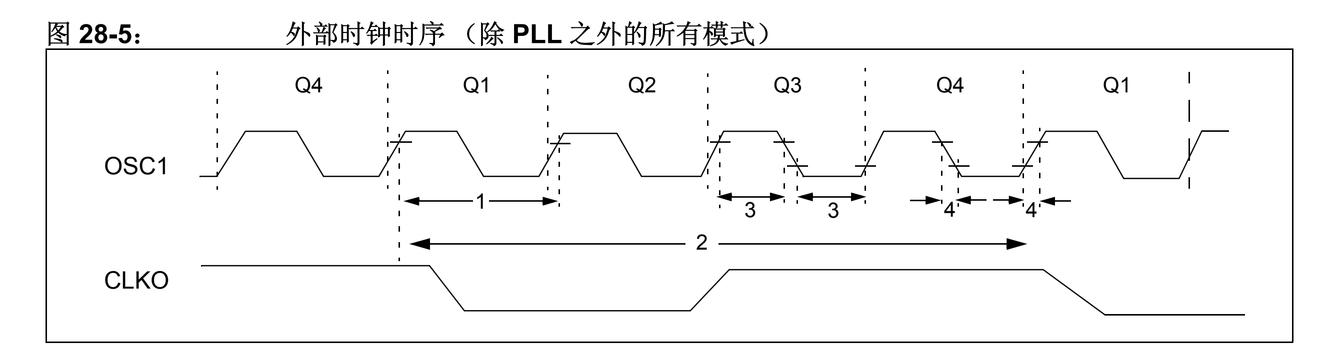


表 **28-8**： 外部时钟时序要求

参数编号	符号	特性	最小值	最大值	单位	条件
1A	FOSC	外部 CLKI 频率 (1) 振荡器频率 (1)	DC 0.1 4 4	48 4 25 48	MHz MHz MHz MHz	EC 和 ECIO 振荡器模式 XT 和 XTPLL 振荡器模式 HS 振荡器模式 HSPLL 振荡器模式
1	TOSC	外部时钟 CLKI 周期 (1) 振荡器周期 (1)	20.8 250 25 20.8	— 10,000 250 250	ns ns ns ns	EC 和 ECIO 振荡器模式 XT 振荡器模式 HS 振荡器模式 HSPLL 振荡器模式
2	TCY	指令周期时间 (1)	83.3	—	ns	TCY = 4/FOSC
3	TosL， TosH	外部时钟输入 （OSC1） 的高电平或低电平时间	30 10	— —	ns ns	XT 振荡器模式 HS 振荡器模式
4	TosR， TosF	外部时钟输入 （OSC1）的 上升或下降时间	— —	20 7.5	ns ns	XT 振荡器模式 HS 振荡器模式

注 **1**： 对于除 PLL 的所有配置来说，指令周期时间 （TCY）等于输入振荡器时基周期的 4 倍。所有值均为在特定的振荡器模式下，器件在标准工作条件下执行代码时获得的特征数据。超过规定值可能导致振荡器运行不稳定和 / 或电流消耗超出预期值。所有器件在测试 “最小值”时，都在 OSC1/CLKI 引脚连接了外部时钟。当使用了外部时钟输入时，所有器件的 “最大”周期时间限制为 “DC”（没有时钟）。

 初稿

表 28-9： PLL 时钟时序规范 （VDD = 4.2V 至 5.5V）

参数编号	符号	特性	最小值	典型值 †	最大值	单位	条件
F10	FOSC	振荡器频率范围	4	—	48	MHz	
F11	FSYS	片上 VCO 系统频率	—	96	—	MHz	
F12	t_{rc}	PLL 起振时间 （锁定时间）	—	—	2	ms	
F13	ΔCLK	CLKO 稳定性 （抗抖动）	-0.25	—	+0.25	%	

† “典型值”栏中的数据均为在 5V， 25°C 下的值。这些参数仅作为设计参考，未经测试。

表 28-10： 交流规范： 内部 RC 精度
PIC18F2455/2550/4455/4550 （工业级）
PIC18LF2455/2550/4455/4550 （工业级）

PIC18LF2455/2550/4455/4550 （工业级）		标准工作条件 （除非另外声明） 工作温度 -40°C ≤ TA ≤ +85°C （工业级）					
PIC18F2455/2550/4455/4550 （工业级）		标准工作条件 （除非另外声明） 工作温度 -40°C ≤ TA ≤ +85°C （工业级）					
参数编号	器件	最小值	典型值	最大值	单位	条件	
	在频率为 8 MHz、 4 MHz、 2 MHz、 1 MHz、 500 kHz、 250 kHz 和 125 kHz 时的 INTOSC 精度 [1]						
	PIC18LF2455/2550/4455/4550	-2	+/-1	2	%	+25°C	VDD = 2.7-3.3V
		-5	—	5	%	-10°C 至 +85°C	VDD = 2.7-3.3V
		-10	+/-1	10	%	-40°C 至 +85°C	VDD = 2.7-3.3V
	PIC18F2455/2550/4455/4550	-2	+/-1	2	%	+25°C	VDD = 4.5-5.5V
		-5	—	5	%	-10°C 至 +85°C	VDD = 4.5-5.5V
		-10	+/-1	10	%	-40°C 至 +85°C	VDD = 4.5-5.5V
	频率为 31 kHz 时的 INTRC 精度 [2]						
	PIC18LF2455/2550/4455/4550	26.562	—	35.938	kHz	-40°C 至 +85°C	VDD = 2.7-3.3V
	PIC18F2455/2550/4455/4550	26.562	—	35.938	kHz	-40°C 至 +85°C	VDD = 4.5-5.5V

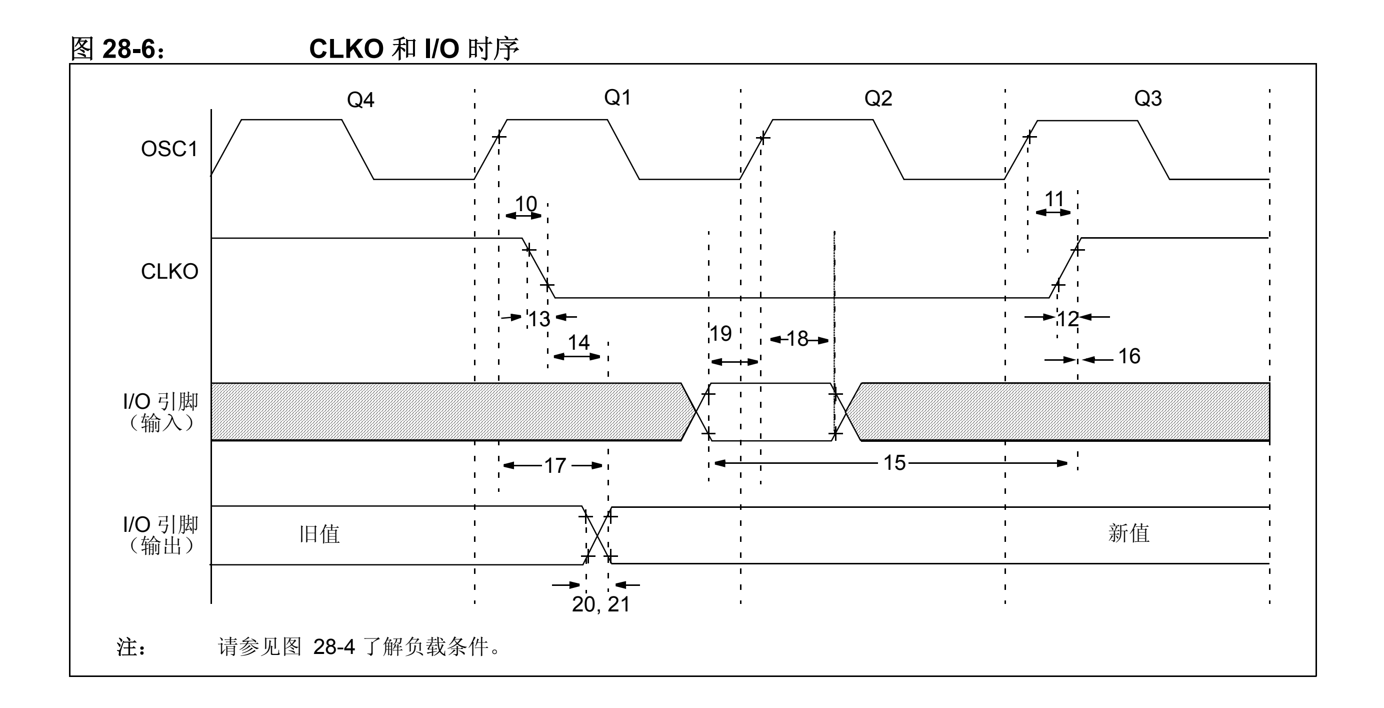
图注： 阴影行是为了增强表的可读性。

注 1： 频率校准温度为 25°C。 OSCTUNE 寄存器可用于补偿温度漂移。

2： 校准后的 INTRC 频率。

3： INTRC 频率随 VDD 的改变而变化。

图 28-6： CLKO 和 I/O 时序



注： 请参见图 28-4 了解负载条件。

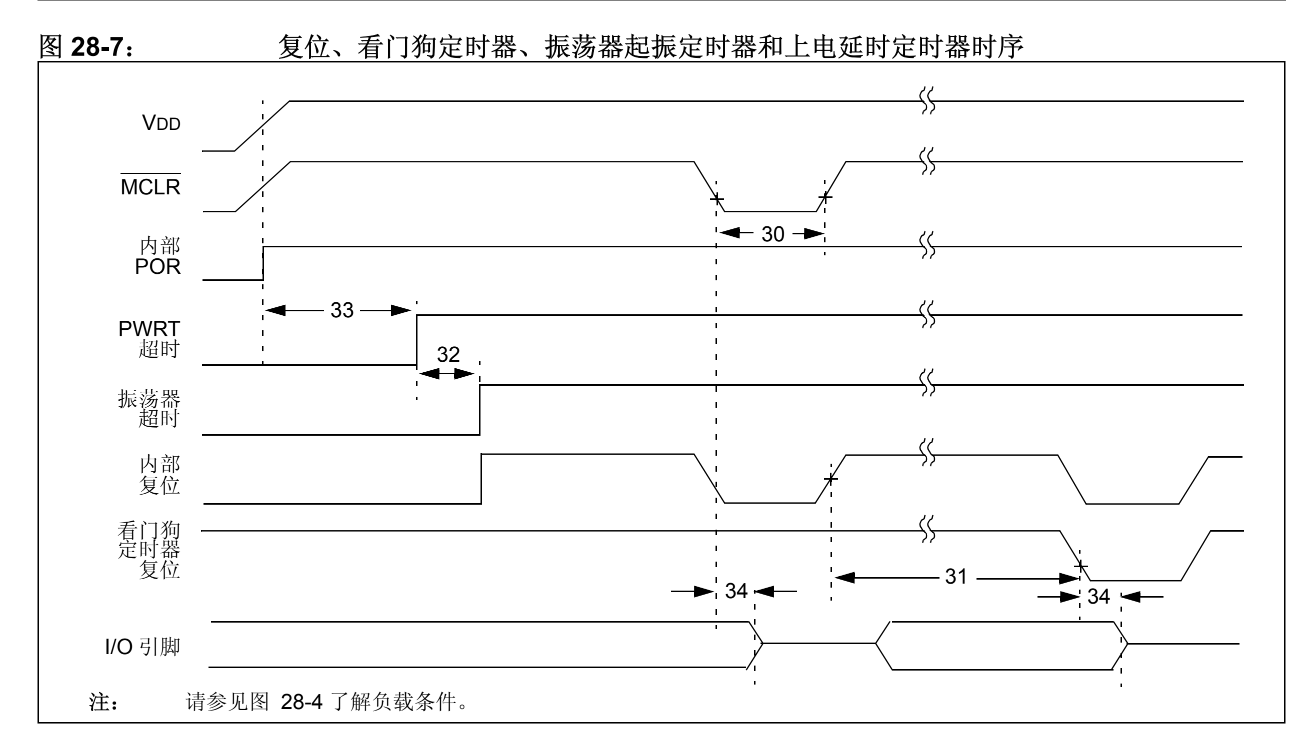
表 28-11： CLKO 和 I/O 时序要求

参数编号	符号	特性		最小值	典型值	最大值	单位	条件
10	TosH2ckL	OSC1 ↑到 CLKO ↓的时间		—	75	200	ns	（注 **1**）
11	TosH2ckH	OSC1 ↑到 CLKO ↑的时间		—	75	200	ns	（注 **1**）
12	TckR	CLKO 上升时间		—	35	100	ns	（注 **1**）
13	TckF	CLKO 下降时间		—	35	100	ns	（注 **1**）
14	TckL2ioV	CLKO ↓至端口输出有效的时间		—	—	0.5 TCY + 20	ns	（注 **1**）
15	TioV2ckH	在 CLKO ↑前端口输入有效时间		0.25 TCY + 25	—	—	ns	（注 **1**）
16	TckH2ioI	在 CLKO ↑后端口输入保持时间		0	—	—	ns	（注 **1**）
17	TosH2ioV	OSC1 ↑ （Q1 周期）至端口输出有效的时间		—	50	150	ns	
18	TosH2ioI	OSC1 ↑ （Q2 周期）至端口输入无效的时间（I/O 输入保持时间）	PIC18**F**XXXX	100	—	—	ns	
18A			PIC18**LF**XXXX	200	—	—	ns	VDD = 2.0V
19	TioV2osH	端口输入有效至 OSC1 ↑的时间 （I/O 输入建立时间）		0	—	—	ns	
20	TioR	端口输出上升时间	PIC18**F**XXXX	—	10	25	ns	
20A			PIC18**LF**XXXX	—	—	60	ns	VDD = 2.0V
21	TioF	端口输出下降时间	PIC18**F**XXXX	—	10	25	ns	
21A			PIC18**LF**XXXX	—	—	60	ns	VDD = 2.0V
22†	TINP	INT 引脚高电平或低电平时间		TCY	—	—	ns	
23†	TRBP	RB7:RB4 电平变化中断信号的高电平或低电平时间		TCY	—	—	ns	

† 这些参数是与内部时钟边沿无关的异步事件。

注 **1**： 测量是在 RC 模式下进行的，其中 CLKO 输出为 4 x TOSC。

图 28-7： 复位、看门狗定时器、振荡器起振定时器和上电延时定时器时序



注： 请参见图 28-4 了解负载条件。

图 28-8： 欠压复位时序

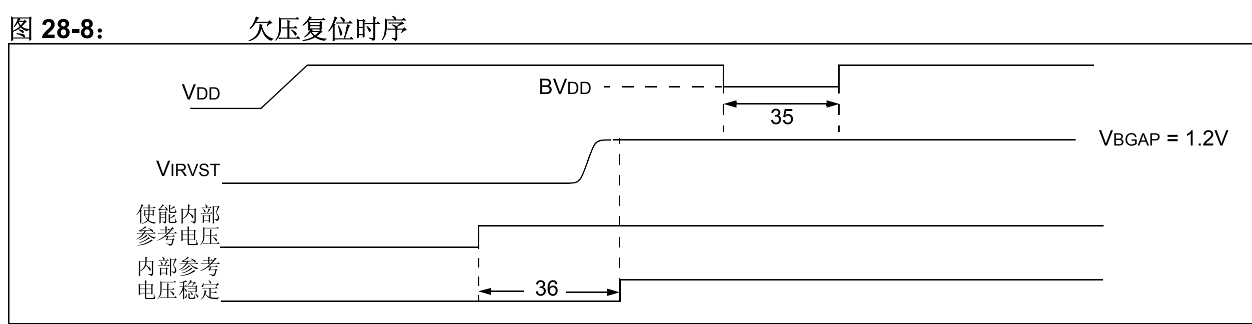
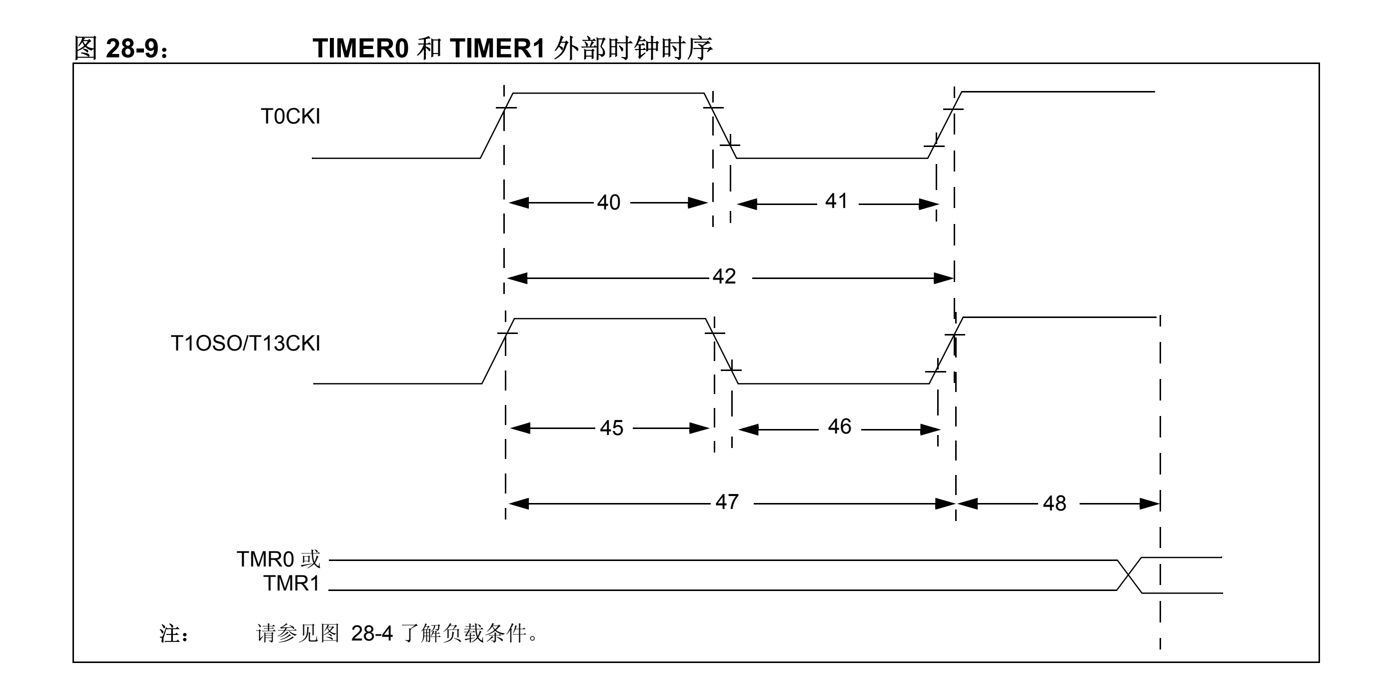


表 28-12： 复位、看门狗定时器、振荡器起振定时器、上电延时定时器和欠压复位要求

参数编号	符号	特性	最小值	典型值	最大值	单位	条件
30	TmcL	MCLR 脉冲宽度 （低电平）	2	—	—	μs	
31	TWDT	看门狗定时器超时溢出周期 （无后分频器）	—	4.00	TBD	ms	
32	TOST	振荡器起振定时器周期	1024 TOSC	—	1024 TOSC	—	TOSC = OSC1 周期
33	TPWRT	上电延时定时器周期	—	65.5	TBD	ms	
34	TIOZ	自 MCLR 低电平或看门狗定时器复位起 I/O 处于高阻态的时间	—	2	—	μs	
35	TBOR	欠压复位脉冲宽度	200	—	—	μs	VDD ≤ BVDD （见 D005）
36	TIRVST	内部参考电压稳定时间	—	20	50	μs	
37	TLVD	高 / 低压检测脉冲宽度	200	—	—	μs	VDD ≤ VLVD
38	TCSD	CPU 的启动时间	5	—	10	μs	
39	TIOBST	INTOSC 电路稳定时间	—	1	—	ms	

图注： TBD = 待定

图 28-9：　TIMER0 和 TIMER1 外部时钟时序



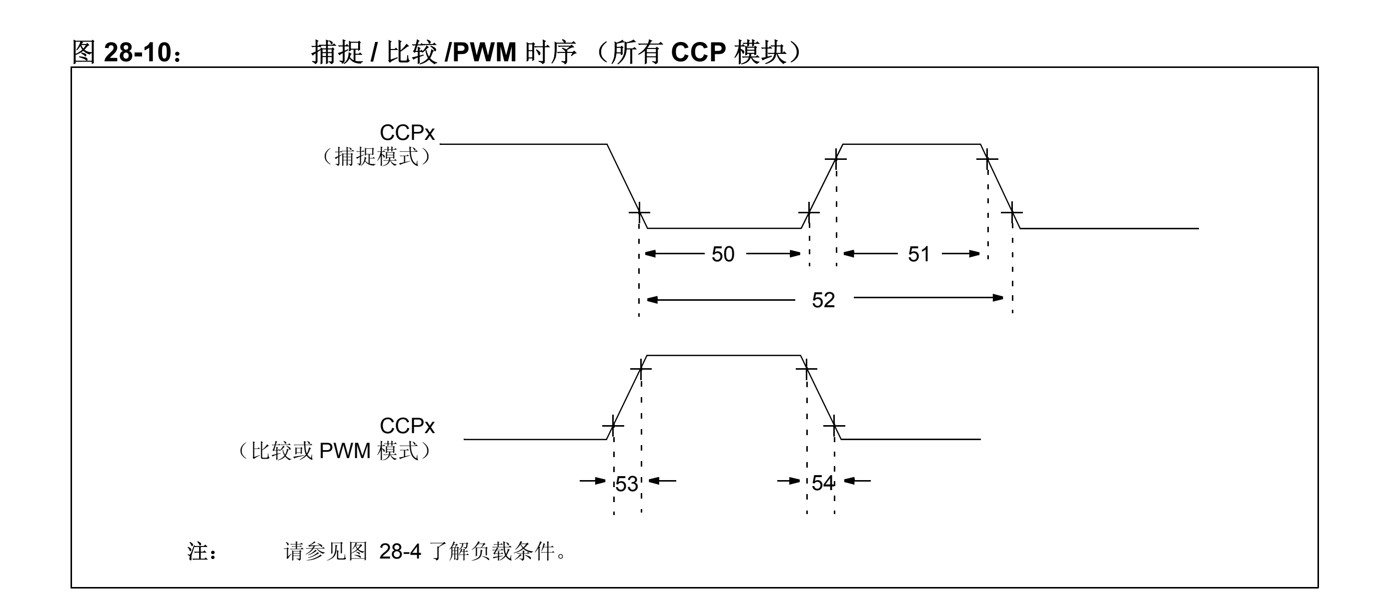
注：　请参见图 28-4 了解负载条件。

表 28-13：　TIMER0 和 TIMER1 外部时钟要求

参数编号	符号	特性			最小值	最大值	单位	条件
40	Tt0H	T0CKI 高电平脉冲宽度		无预分频器	0.5 TCY + 20	—	ns	
				有预分频器	10	—	ns	
41	Tt0L	T0CKI 低电平脉冲宽度		无预分频器	0.5 TCY + 20	—	ns	
				有预分频器	10	—	ns	
42	Tt0P	T0CKI 周期		无预分频器	TCY +10	—	ns	
				有预分频器	取较大值：20 ns 或 (TCY + 40)/N	—	ns	N = 预分频值（1、2、4... 256）
45	Tt1H	T13CKI 高电平时间	同步，无预分频器		0.5 TCY + 20	—	ns	
			同步，有预分频器	PIC18**F**XXXX	10	—	ns	
				PIC18**LF**XXXX	25	—	ns	VDD = 2.0V
			异步	PIC18**F**XXXX	30	—	ns	
				PIC18**LF**XXXX	50	—	ns	VDD = 2.0V
46	Tt1L	T13CKI 低电平时间	同步，无预分频器		0.5 TCY +5	—	ns	
			同步，有预分频器	PIC18**F**XXXX	10	—	ns	
				PIC18**LF**XXXX	25	—	ns	VDD = 2.0V
			异步	PIC18**F**XXXX	30	—	ns	
				PIC18**LF**XXXX	50	—	ns	VDD = 2.0V
47	Tt1P	T13CKI 输入周期	同步		取较大值：20 ns 或 (TCY + 40)/N	—	ns	N = 预分频值（1、2、4 或 8）
			异步		60	—	ns	
	Ft1	T13CKI 振荡器输入频率范围			DC	50	kHz	
48	Tcke2tmrl	从外部 T13CKI 时钟边沿到定时器加 1 的延时			2 TOSC	7 TOSC	—	

　初稿　

图 28-10： 捕捉 / 比较 /PWM 时序 （所有 CCP 模块）



注： 请参见图 28-4 了解负载条件。

表 28-14： 捕捉 / 比较 /PWM 要求 （所有 CCP 模块）

参数编号	符号	特性			最小值	最大值	单位	条件
50	TccL	CCPx 输入低电平时间	无预分频器		0.5 TCY + 20	—	ns	
			有预分频器	PIC18**F**XXXX	10	—	ns	
				PIC18**LF**XXXX	20	—	ns	VDD = 2.0V
51	TccH	CCPx 输入高电平时间	无预分频器		0.5 TCY + 20	—	ns	
			有预分频器	PIC18**F**XXXX	10	—	ns	
				PIC18**LF**XXXX	20	—	ns	VDD = 2.0V
52	TccP	CCPx 输入周期			(3 TCY + 40) / N	—	ns	N = 预分频值（1、 4 或 16）
53	TccR	CCPx 输出上升时间		PIC18**F**XXXX	—	25	ns	
				PIC18**LF**XXXX	—	45	ns	VDD = 2.0V
54	TccF	CCPx 输出下降时间		PIC18**F**XXXX	—	25	ns	
				PIC18**LF**XXXX	—	45	ns	VDD = 2.0V

图 28-11： SPI 主控模式时序示例 （CKE = 0）

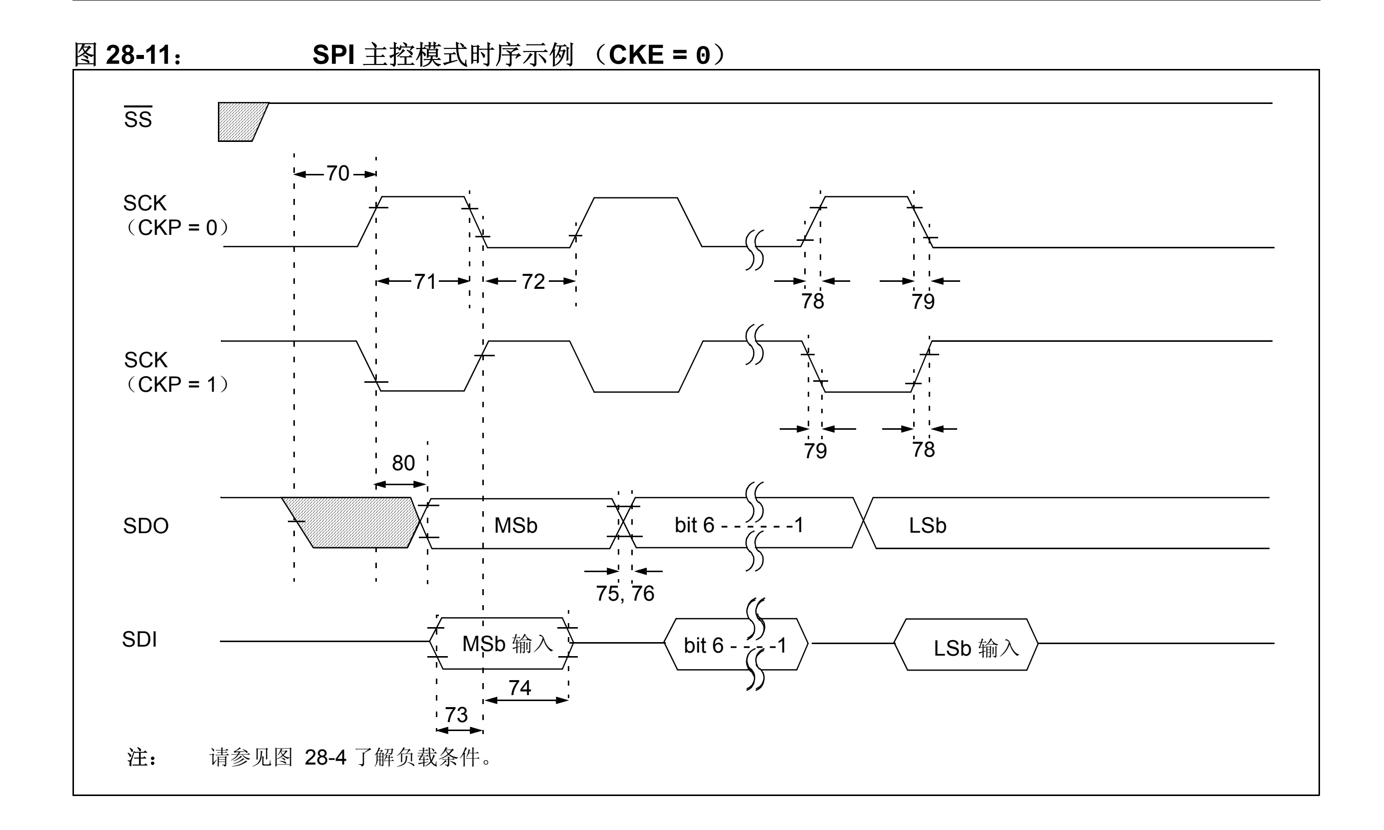
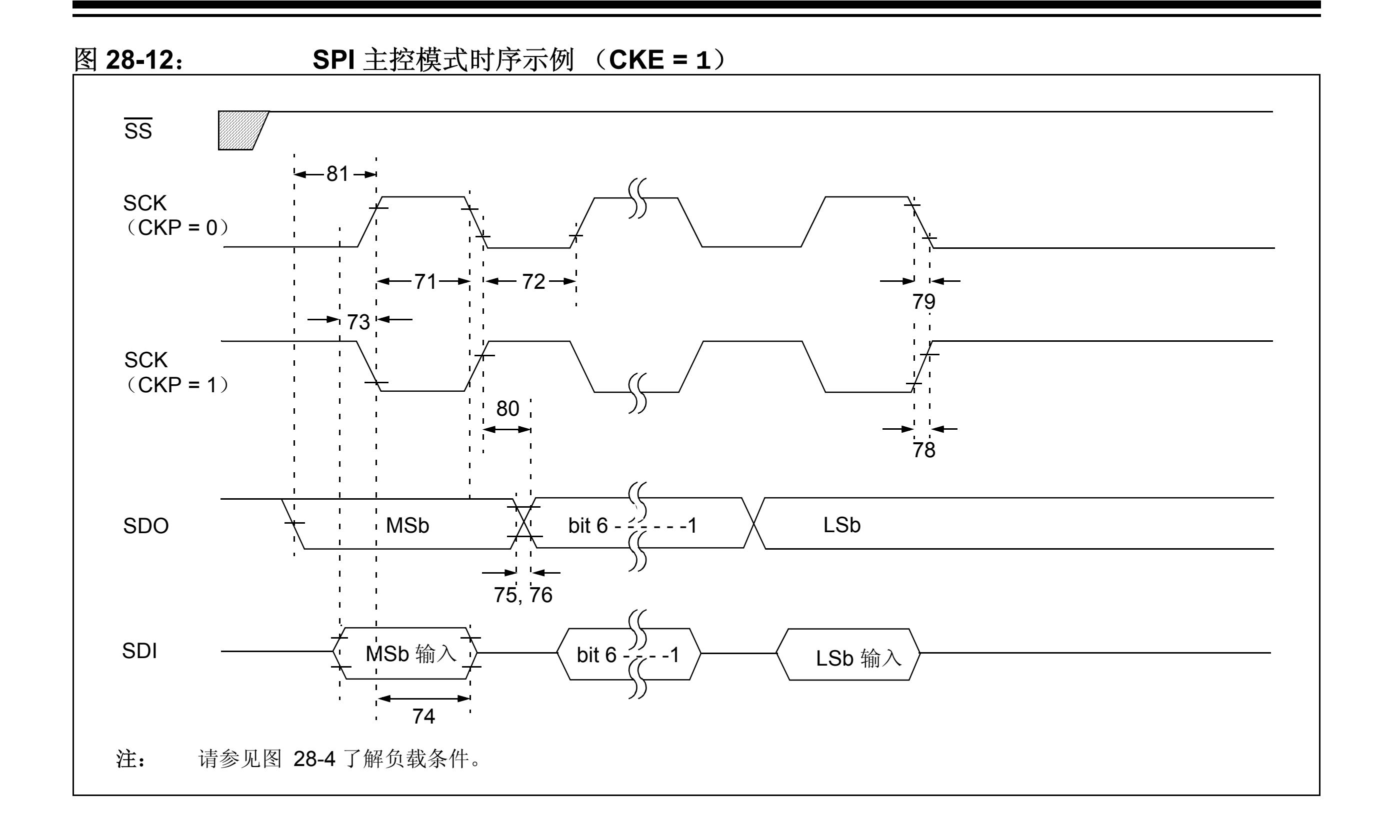


表 28-15： SPI 主控模式要求示例 （CKE = 0）

参数编号	符号	特性		最小值	最大值	单位	条件
70	TssL2scH，TssL2scL	SS ↓ 至 SCK ↓ 或 SCK ↑ 输入的时间		TCY	—	ns	
71	TscH	SCK 输入高电平时间（从动模式）	连续	1.25 TCY + 30	—	ns	
71A			单字节	40	—	ns	（注 1）
72	TscL	SCK 输入低电平时间（从动模式）	连续	1.25 TCY + 30	—	ns	
72A			单字节	40	—	ns	（注 1）
73	TdiV2scH，TdiV2scL	SDI 数据输入到 SCK 边沿的建立时间		100	—	ns	
73A	Tb2b	字节 1 的最后一个时钟边沿到字节 2 的第一个时钟边沿的时间		1.5 TCY +40	—	ns	（注 2）
74	TscH2diL，TscL2diL	SDI 数据输入到 SCK 边沿的保持时间		100	—	ns	
75	TdoR	SDO 数据输出上升时间	PIC18**F**XXXX	—	25	ns	
			PIC18**LF**XXXX	—	45	ns	VDD = 2.0V
76	TdoF	SDO 数据输出下降时间		—	25	ns	
78	TscR	SCK 输出上升时间（主控模式）	PIC18**F**XXXX	—	25	ns	
			PIC18**LF**XXXX	—	45	ns	VDD = 2.0V
79	TscF	SCK 输出下降时间 （主控模式）		—	25	ns	
80	TscH2doV，TscL2doV	SCK 边沿后 SDO 数据输出有效时间	PIC18**F**XXXX	—	50	ns	
			PIC18**LF**XXXX	—	100	ns	VDD = 2.0V

注 **1**： 要求使用参数 73A。
2： 仅当使用参数 71A 和 72A 时。

图 28-12： SPI 主控模式时序示例 （CKE = 1）



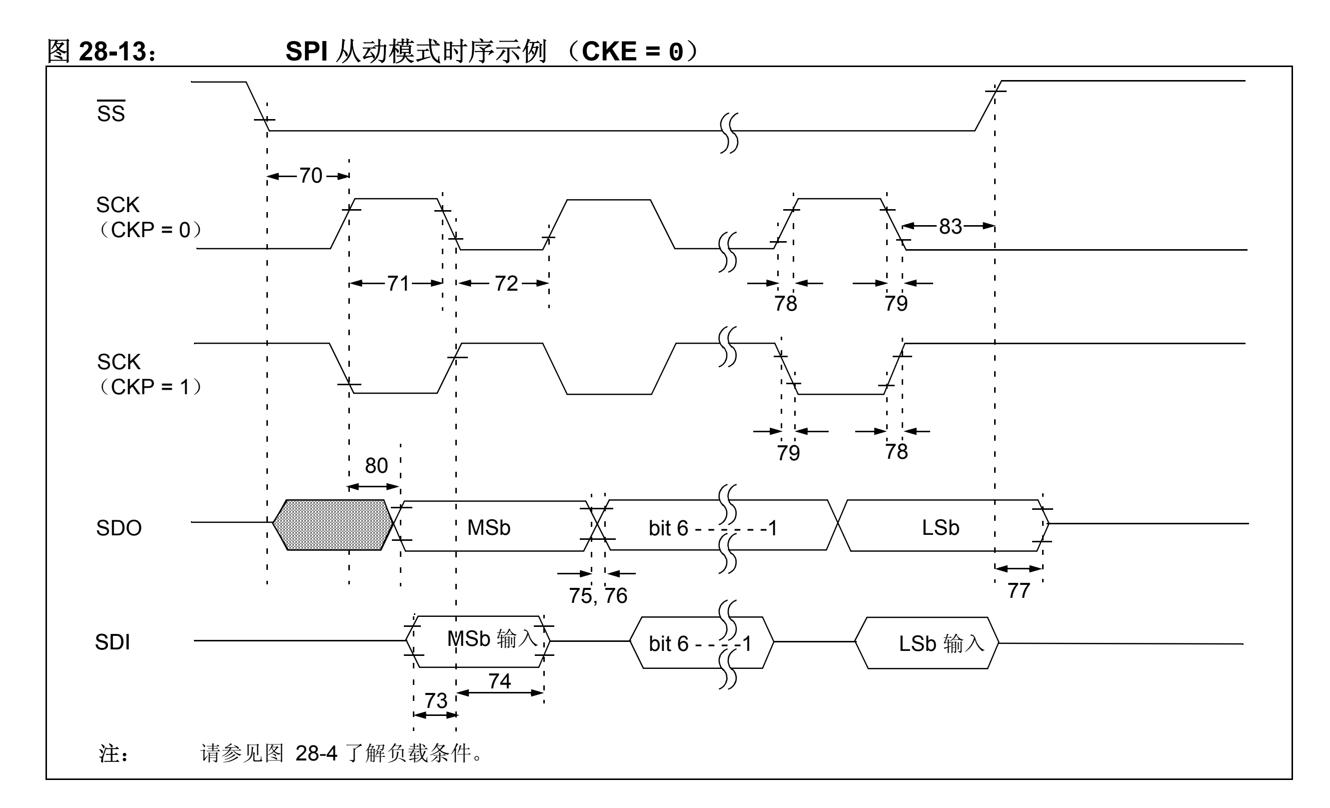
注： 请参见图 28-4 了解负载条件。

表 28-16： SPI 主控模式要求示例 （CKE = 1）

参数编号	符号	特性		最小值	最大值	单位	条件
71	TscH	SCK 输入高电平时间（从动模式）	连续	1.25 TCY + 30	—	ns	
71A			单字节	40	—	ns	（注 1）
72	TscL	SCK 输入低电平时间（从动模式）	连续	1.25 TCY + 30	—	ns	
72A			单字节	40	—	ns	（注 1）
73	TdiV2scH，TdiV2scL	SDI 数据输入到 SCK 边沿的建立时间		100	—	ns	
73A	Tb2b	字节 1 的最后一个时钟边沿到字节 2 的第一个时钟边沿的时间		1.5 TCY +40	—	ns	（注 2）
74	TscH2diL，TscL2diL	SDI 数据输入到 SCK 边沿的保持时间		100	—	ns	
75	TdoR	SDO 数据输出上升时间	PIC18**F**XXXX	—	25	ns	
			PIC18**LF**XXXX		45	ns	VDD = 2.0V
76	TdoF	SDO 数据输出下降时间		—	25	ns	
78	TscR	SCK 输出上升时间（主控模式）	PIC18**F**XXXX	—	25	ns	
			PIC18**LF**XXXX		45	ns	VDD = 2.0V
79	TscF	SCK 输出下降时间 （主控模式）		—	25	ns	
80	TscH2doV，TscL2doV	SCK 边沿后 SDO 数据输出有效的时间	PIC18**F**XXXX	—	50	ns	
			PIC18**LF**XXXX		100	ns	VDD = 2.0V
81	TdoV2scH，TdoV2scL	SDO 数据输出到 SCK 边沿的建立时间		TCY	—	ns	

注 **1**： 要求使用参数 73A。
2： 仅当使用参数 71A 和 72A 时。

图 28-13： SPI 从动模式时序示例 （CKE = 0）



注： 请参见图 28-4 了解负载条件。

表 28-17： SPI 从动模式要求示例 （CKE = 0）

参数编号	符号	特性		最小值	最大值	单位	条件
70	TssL2scH，TssL2scL	SS ↓ 至 SCK ↓ 或 SCK ↑ 输入的时间		TCY	—	ns	
71	TscH	SCK 输入高电平时间 （从动模式）	连续	1.25 TCY + 30	—	ns	
71A			单字节	40	—	ns	（注 **1**）
72	TscL	SCK 输入低电平时间 （从动模式）	连续	1.25 TCY + 30	—	ns	
72A			单字节	40	—	ns	（注 **1**）
73	TdiV2scH，TdiV2scL	SDI 数据输入到 SCK 边沿的建立时间		100	—	ns	
73A	Tb2b	字节 1 的最后一个时钟边沿到字节 2 的第一个时钟边沿的时间		1.5 TCY +40	—	ns	（注 **2**）
74	TscH2diL，TscL2diL	SDI 数据输入到 SCK 边沿的保持时间		100	—	ns	
75	TdoR	SDO 数据输出上升时间	PIC18**F**XXXX	—	25	ns	
			PIC18**LF**XXXX		45	ns	VDD = 2.0V
76	TdoF	SDO 数据输出下降时间		—	25	ns	
77	TssH2doZ	SS ↑ 到 SDO 输出高阻态的时间		10	50	ns	
78	TscR	SCK 输出上升时间 （主控模式）	PIC18**F**XXXX	—	25	ns	
			PIC18**LF**XXXX		45	ns	VDD = 2.0V
79	TscF	SCK 输出下降时间 （主控模式）		—	25	ns	
80	TscH2doV，TscL2doV	SCK 边沿后 SDO 数据输出有效的时间	PIC18**F**XXXX	—	50	ns	
			PIC18**LF**XXXX		100	ns	VDD = 2.0V
83	TscH2ssH，TscL2ssH	SCK 边沿之后到 SS ↑ 的时间		1.5 TCY + 40	—	ns	

注 **1：** 要求使用参数 73A。
2： 仅当使用参数 71A 和 72A 时。

图 28-14： SPI 从动模式时序示例 （CKE = 1）

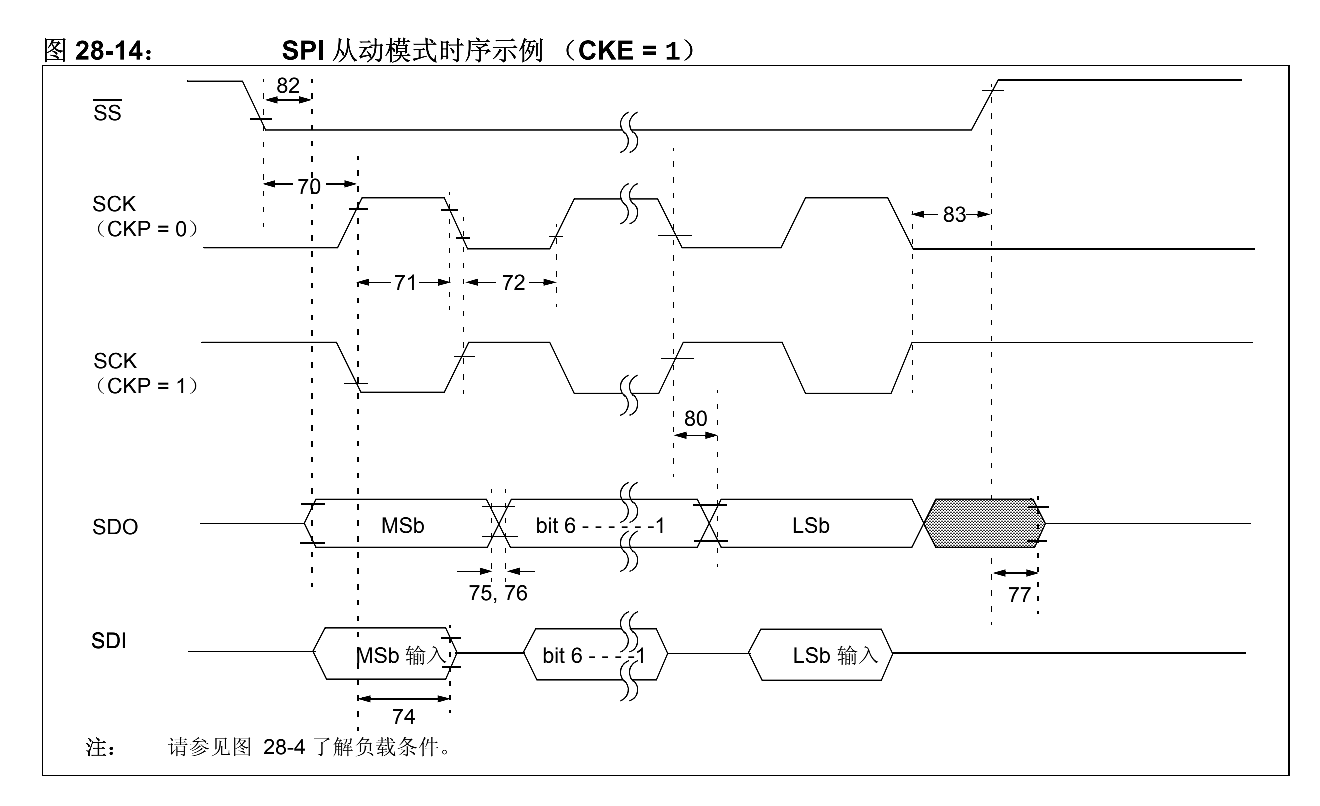


表 28-18： SPI 从动模式要求示例 （CKE = 1）

参数编号	符号	特性		最小值	最大值	单位	条件
70	TssL2scH，TssL2scL	SS ↓至 SCK ↓或 SCK ↑输入的时间		TCY	—	ns	
71	TscH	SCK 输入高电平时间（从动模式）	连续	1.25 TCY + 30	—	ns	
71A			单字节	40	—	ns	（注 1）
72	TscL	SCK 输入低电平时间（从动模式）	连续	1.25 TCY + 30	—	ns	
72A			单字节	40	—	ns	（注 1）
73A	Tb2b	字节 1 的最后一个时钟边沿到字节 2 的第一个时钟边沿的时间		1.5 TCY +40	—	ns	（注 2）
74	TscH2diL，TscL2diL	SDI 数据输入到 SCK 边沿的保持时间		100	—	ns	
75	TdoR	SDO 数据输出上升时间	PIC18**F**XXXX	—	25	ns	
			PIC18**LF**XXXX		45	ns	VDD = 2.0V
76	TdoF	SDO 数据输出下降时间		—	25	ns	
77	TssH2doZ	SS ↑到 SDO 输出高阻态的时间		10	50	ns	
78	TscR	SCK 输出上升时间（主控模式）	PIC18**F**XXXX	—	25	ns	
			PIC18**LF**XXXX	—	45	ns	VDD = 2.0V
79	TscF	SCK 输出下降时间 （主控模式）		—	25	ns	
80	TscH2doV，TscL2doV	SCK 边沿后 SDO 数据输出有效的时间	PIC18**F**XXXX	—	50	ns	
			PIC18**LF**XXXX	—	100	ns	VDD = 2.0V
82	TssL2doV	在 SS ↓边沿之后 SDO 数据输出有效的时间	PIC18**F**XXXX	—	50	ns	
			PIC18**LF**XXXX	—	100	ns	VDD = 2.0V
83	TscH2ssH，TscL2ssH	SCK 边沿之后出现 SS ↑的时间		1.5 TCY +40	—	ns	

注 **1：** 要求使用参数 73A。
2： 仅当使用参数 71A 和 72A 时。

图 28-15： I²C™ 总线启动 / 停止位时序

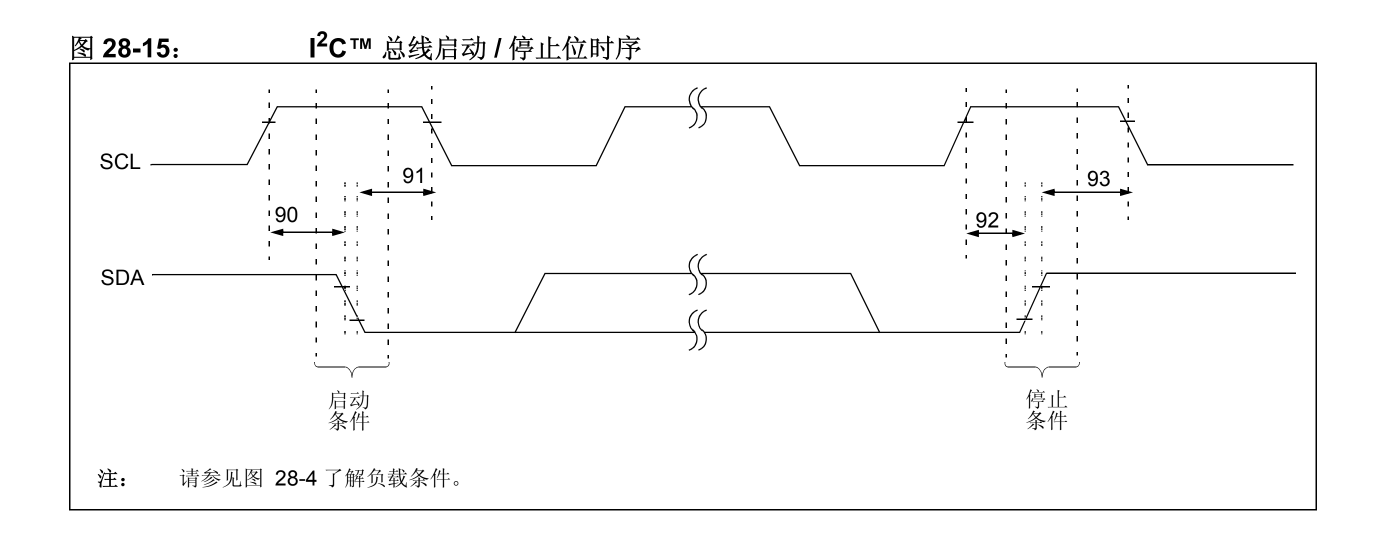
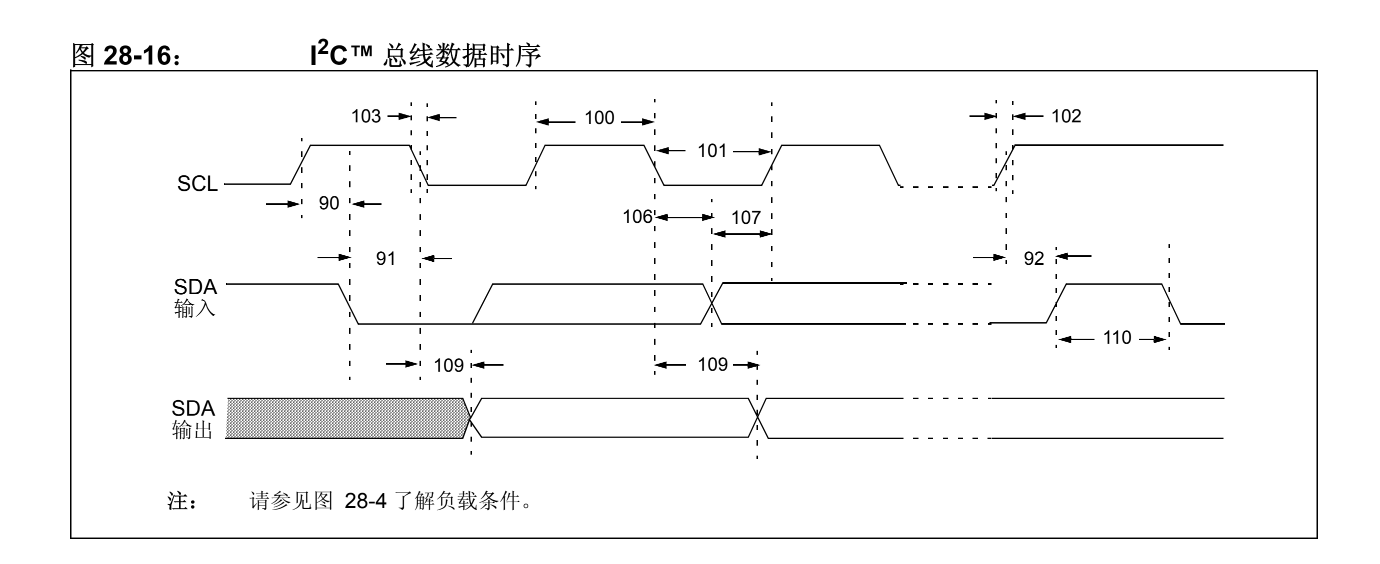


表 28-19： I²C™ 总线启动 / 停止位要求 （从动模式）

参数编号	符号	特性		最小值	最大值	单位	条件
90	TSU:STA	启动条件建立时间	100 kHz 模式	4700	—	ns	仅与重复启动条件相关
			400 kHz 模式	600	—		
91	THD:STA	启动条件保持时间	100 kHz 模式	4000	—	ns	这个周期后产生第一个时钟脉冲
			400 kHz 模式	600	—		
92	TSU:STO	停止条件建立时间	100 kHz 模式	4700	—	ns	
			400 kHz 模式	600	—		
93	THD:STO	停止条件保持时间	100 kHz 模式	4000	—	ns	
			400 kHz 模式	600	—		

图 28-16： I²C™ 总线数据时序



 初稿

表 28-20： I²C™ 总线数据要求 （从动模式）

参数编号	符号	特性		最小值	最大值	单位	条件
100	THIGH	时钟高电平时间	100 kHz 模式	4.0	—	μs	PIC18FXXXX 的工作频率至少为 1.5 MHz
			400 kHz 模式	0.6	—	μs	PIC18FXXXX 的工作频率至少为 10 MHz
			SSP 模块	1.5 TCY	—		
101	TLOW	时钟低电平时间	100 kHz 模式	4.7	—	μs	PIC18FXXXX 的工作频率至少为 1.5 MHz
			400 kHz 模式	1.3	—	μs	PIC18FXXXX 的工作频率至少为 10 MHz
			SSP 模块	1.5 TCY	—		
102	TR	SDA 和 SCL 上升时间	100 kHz 模式	—	1000	ns	
			400 kHz 模式	20 + 0.1 CB	300	ns	CB 值的范围在 10 到 400 pF 之间
103	TF	SDA 和 SCL 下降时间	100 kHz 模式	—	300	ns	
			400 kHz 模式	20 + 0.1 CB	300	ns	CB 值的范围在 10 到 400 pF 之间
90	TSU:STA	启动条件建立时间	100 kHz 模式	4.7	—	μs	仅与重复启动条件相关
			400 kHz 模式	0.6	—	μs	
91	THD:STA	启动条件保持时间	100 kHz 模式	4.0	—	μs	这个周期后产生第一个时钟脉冲
			400 kHz 模式	0.6	—	μs	
106	THD:DAT	数据输入保持时间	100 kHz 模式	0	—	ns	
			400 kHz 模式	0	0.9	μs	
107	TSU:DAT	数据输入建立时间	100 kHz 模式	250	—	ns	（注 **2**）
			400 kHz 模式	100	—	ns	
92	TSU:STO	停止条件建立时间	100 kHz 模式	4.7	—	μs	
			400 kHz 模式	0.6	—	μs	
109	TAA	时钟输出有效时间	100 kHz 模式	—	3500	ns	（注 **1**）
			400 kHz 模式	—	—	ns	
110	TBUF	总线空闲时间	100 kHz 模式	4.7	—	μs	在启动一个新的传输前总线必须保持空闲的时间
			400 kHz 模式	1.3	—	μs	
D102	CB	总线的容性负载		—	400	pF	

注 **1**： 为避免意外产生启动或停止条件，作为发送器的器件必须提供此内部最小延迟时间，以覆盖 SCL 下降沿的未定义区域 （最小值 300ns）。

2： 在标准模式的 I²C™ 总线系统中可以使用快速模式的 I²C 总线器件，但必须满足 TSU:DAT ≥ 250 ns 的要求。如果快速模式器件没有延长 SCL 信号的低电平时间，则必然满足此条件。如果该器件延长了 SCL 信号的低电平时间，它必须将下一个数据位输出到 SDA 线， SCL 线被释放前，根据标准模式 I²C 总线规范，TR max. + TSU:DAT = 1000 + 250 = 1250 ns。

图 28-17： 主控 SSP I²C™ 总线启动 / 停止位时序波形

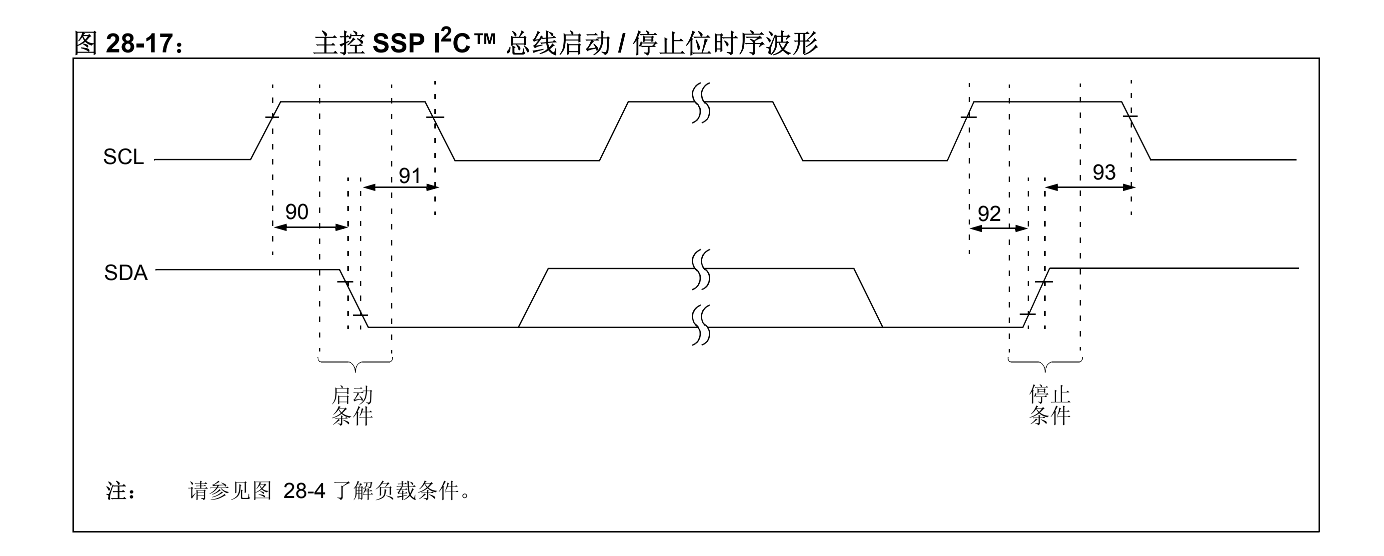
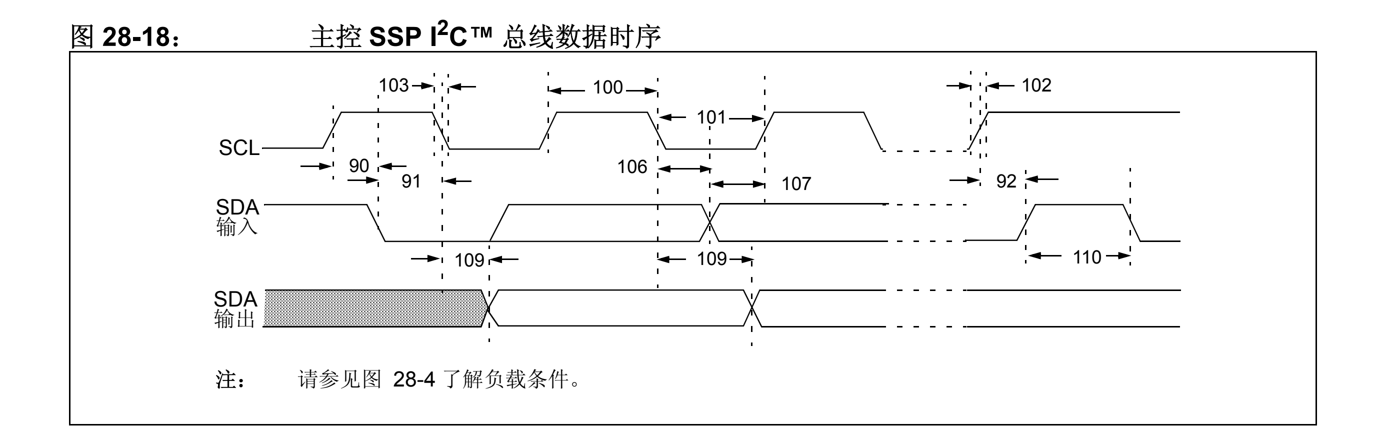


表 28-21： 主控 SSP I²C™ 总线启动 / 停止位要求

参数编号	符号	特性		最小值	最大值	单位	条件
90	TSU:STA	启动条件建立时间	100 kHz 模式	2(TOSC)(BRG + 1)	—	ns	仅与重复启动条件相关
			400 kHz 模式	2(TOSC)(BRG + 1)	—		
			1 MHz 模式 (1)	2(TOSC)(BRG + 1)	—		
91	THD:STA	启动条件保持时间	100 kHz 模式	2(TOSC)(BRG + 1)	—	ns	这个周期后产生第一个时钟脉冲
			400 kHz 模式	2(TOSC)(BRG + 1)	—		
			1 MHz 模式 (1)	2(TOSC)(BRG + 1)	—		
92	TSU:STO	停止条件建立时间	100 kHz 模式	2(TOSC)(BRG + 1)	—	ns	
			400 kHz 模式	2(TOSC)(BRG + 1)	—		
			1 MHz 模式 (1)	2(TOSC)(BRG + 1)	—		
93	THD:STO	停止条件保持时间	100 kHz 模式	2(TOSC)(BRG + 1)	—	ns	
			400 kHz 模式	2(TOSC)(BRG + 1)	—		
			1 MHz 模式 (1)	2(TOSC)(BRG + 1)	—		

注 1： 对于所有 I²C™ 引脚，最大引脚电容为 10 pF。

图 28-18： 主控 SSP I²C™ 总线数据时序



 初稿

表 28-22： 主控 SSP I2C™ 总线数据要求

参数编号	符号	特性		最小值	最大值	单位	条件
100	THIGH	时钟高电平时间	100 kHz 模式	2(TOSC)(BRG + 1)	—	ms	
			400 kHz 模式	2(TOSC)(BRG + 1)	—	ms	
			1 MHz 模式 (1)	2(TOSC)(BRG + 1)	—	ms	
101	TLOW	时钟低电平时间	100 kHz 模式	2(TOSC)(BRG + 1)	—	ms	
			400 kHz 模式	2(TOSC)(BRG + 1)	—	ms	
			1 MHz 模式 (1)	2(TOSC)(BRG + 1)	—	ms	
102	TR	SDA 和 SCL 上升时间	100 kHz 模式	—	1000	ns	CB 值在 10 到 400 pF 之间
			400 kHz 模式	20 + 0.1 CB	300	ns	
			1 MHz 模式 (1)	—	300	ns	
103	TF	SDA 和 SCL 下降时间	100 kHz 模式	—	300	ns	CB 值在 10 到 400 pF 之间
			400 kHz 模式	20 + 0.1 CB	300	ns	
			1 MHz 模式 (1)	—	100	ns	
90	TSU:STA	启动条件建立时间	100 kHz 模式	2(TOSC)(BRG + 1)	—	ms	仅与重复启动条件相关
			400 kHz 模式	2(TOSC)(BRG + 1)	—	ms	
			1 MHz 模式 (1)	2(TOSC)(BRG + 1)	—	ms	
91	THD:STA	启动条件保持时间	100 kHz 模式	2(TOSC)(BRG + 1)	—	ms	这个周期后产生第一个时钟脉冲
			400 kHz 模式	2(TOSC)(BRG + 1)	—	ms	
			1 MHz 模式 (1)	2(TOSC)(BRG + 1)	—	ms	
106	THD:DAT	数据输入保持时间	100 kHz 模式	0	—	ns	
			400 kHz 模式	0	0.9	ms	
107	TSU:DAT	数据输入建立时间	100 kHz 模式	250	—	ns	（注 **2**）
			400 kHz 模式	100	—	ns	
92	TSU:STO	停止条件建立时间	100 kHz 模式	2(TOSC)(BRG + 1)	—	ms	
			400 kHz 模式	2(TOSC)(BRG + 1)	—	ms	
			1 MHz 模式 (1)	2(TOSC)(BRG + 1)	—	ms	
109	TAA	时钟输出有效时间	100 kHz 模式	—	3500	ns	
			400 kHz 模式	—	1000	ns	
			1 MHz 模式 (1)	—	—	ns	
110	TBUF	总线空闲时间	100 kHz 模式	4.7	—	ms	在启动一个新的传输前总线必须保持空闲的时间
			400 kHz 模式	1.3	—	ms	
D102	CB	总线的容性负载		—	400	pF	

注 **1**： 对于所有 I2C™ 引脚，最大引脚电容为 10 pF。

2： 在标准模式的 I2C 总线系统中可以使用快速模式的 I2C 总线器件，但必须满足参数 #107 ≥ 250 ns 的要求。如果快速模式器件没有延长 SCL 信号的低电平时间，则必然满足此条件。如果该器件延长了 SCL 信号的低电平时间，它必须将下一个数据位输出到 SDA 线， SCL 线被释放前，参数 #102 + 参数 #107 = 1000 + 250 = 1250 ns （100 kHz 模式下）。

图 28-19: EUSART 同步发送 (主控 / 从动)时序

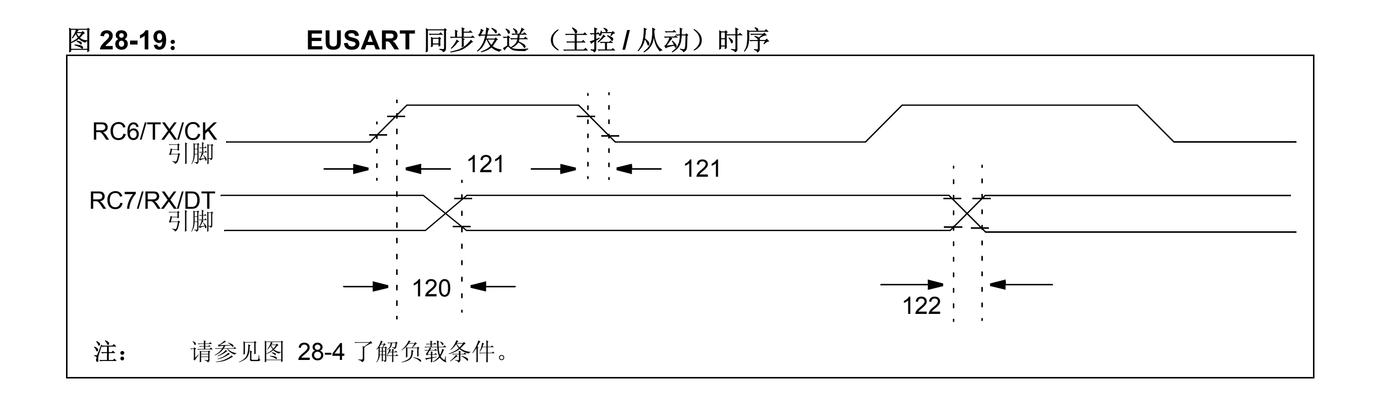


表 28-23: EUSART 同步发送要求

参数编号	符号	特性		最小值	最大值	单位	条件
120	TckH2dtV	同步发送 (主控和从动) 从时钟高电平至数据输出有效的时间	PIC18**F**XXXX	—	40	ns	
			PIC18**LF**XXXX	—	100	ns	VDD = 2.0V
121	Tckrf	时钟输出信号的上升时间和下降时间 (主控模式)	PIC18**F**XXXX	—	20	ns	
			PIC18**LF**XXXX	—	50	ns	VDD = 2.0V
122	Tdtrf	数据输出信号的上升时间和下降时间	PIC18**F**XXXX	—	20	ns	
			PIC18**LF**XXXX	—	50	ns	VDD = 2.0V

图 28-20: EUSART 同步接收 (主控 / 从动)时序

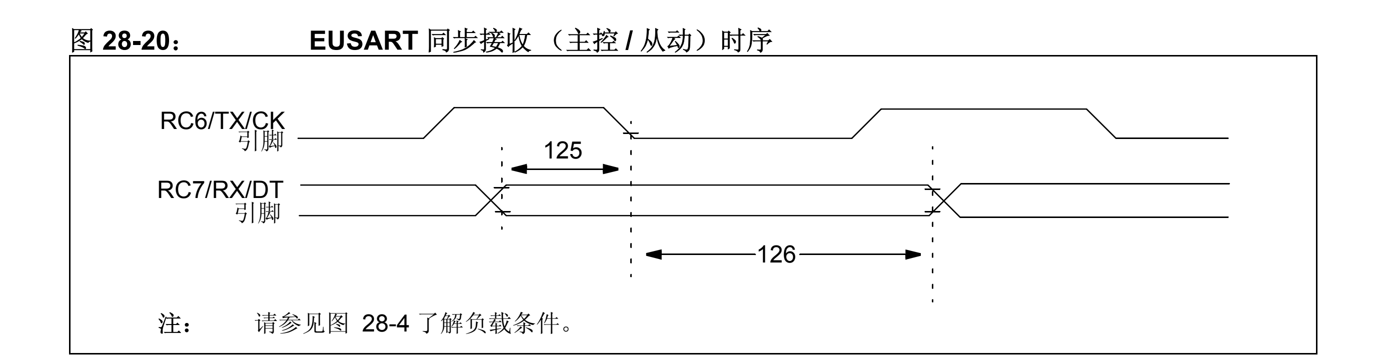


表 28-24: EUSART 同步接收要求

参数编号	符号	特性	最小值	最大值	单位	条件
125	TDTV2CKL	同步接收 (主控和从动) 在 CK ↓之前数据的保持时间 (DT 保持时间)	10	—	ns	
126	TCKL2DTL	在 CK ↓之后数据的保持时间 (DT 保持时间)	15	—	ns	

图 **28-21**： **USB** 信号时序

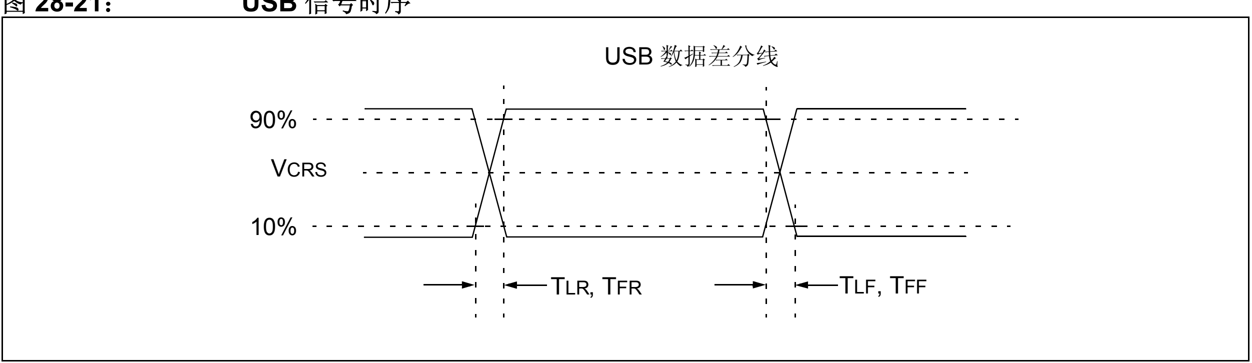


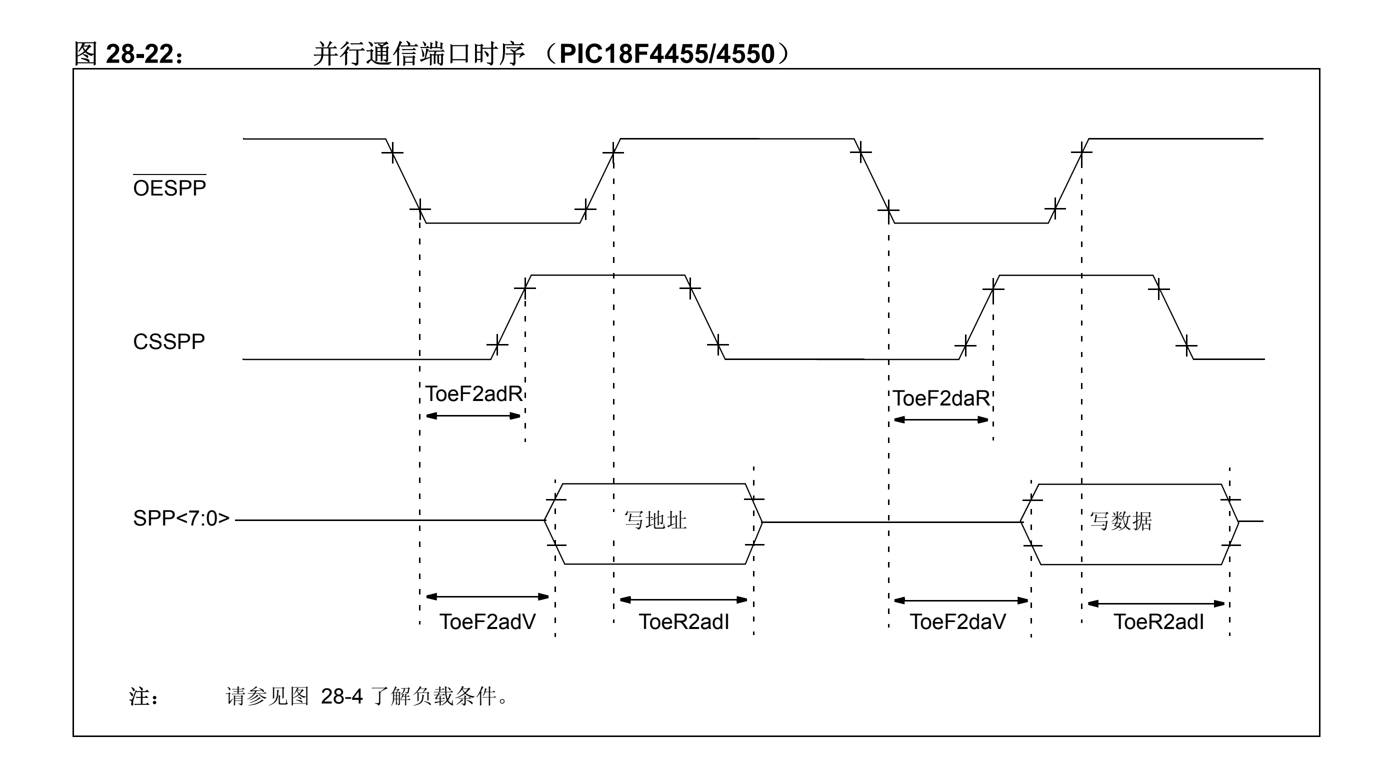
表 **28-25**： **USB** 低速时序要求

参数编号	符号	特性	最小值	典型值	最大值	单位	条件
	TLR	信号上升时间	75	—	300	ns	CL = 200 至 600 pF
	TLF	信号下降时间	75	—	300	ns	CL = 200 至 600 pF
	TLRFM	上升 / 下降时间匹配率	80	—	125	%	

表 **28-26**： **USB** 全速时序要求

参数编号	符号	特性	最小值	典型值	最大值	单位	条件
	TFR	信号上升时间	4	—	20	ns	CL = 50 pF
	TFF	信号下降时间	4	—	20	ns	CL = 50 pF
	TFRFM	上升 / 下降时间匹配率	90	—	111.1	%	

图 28-22： 并行通信端口时序 （PIC18F4455/4550）



注： 请参见图 28-4 了解负载条件。

表 28-27： 并行通信端口要求 （PIC18F4455/4550）

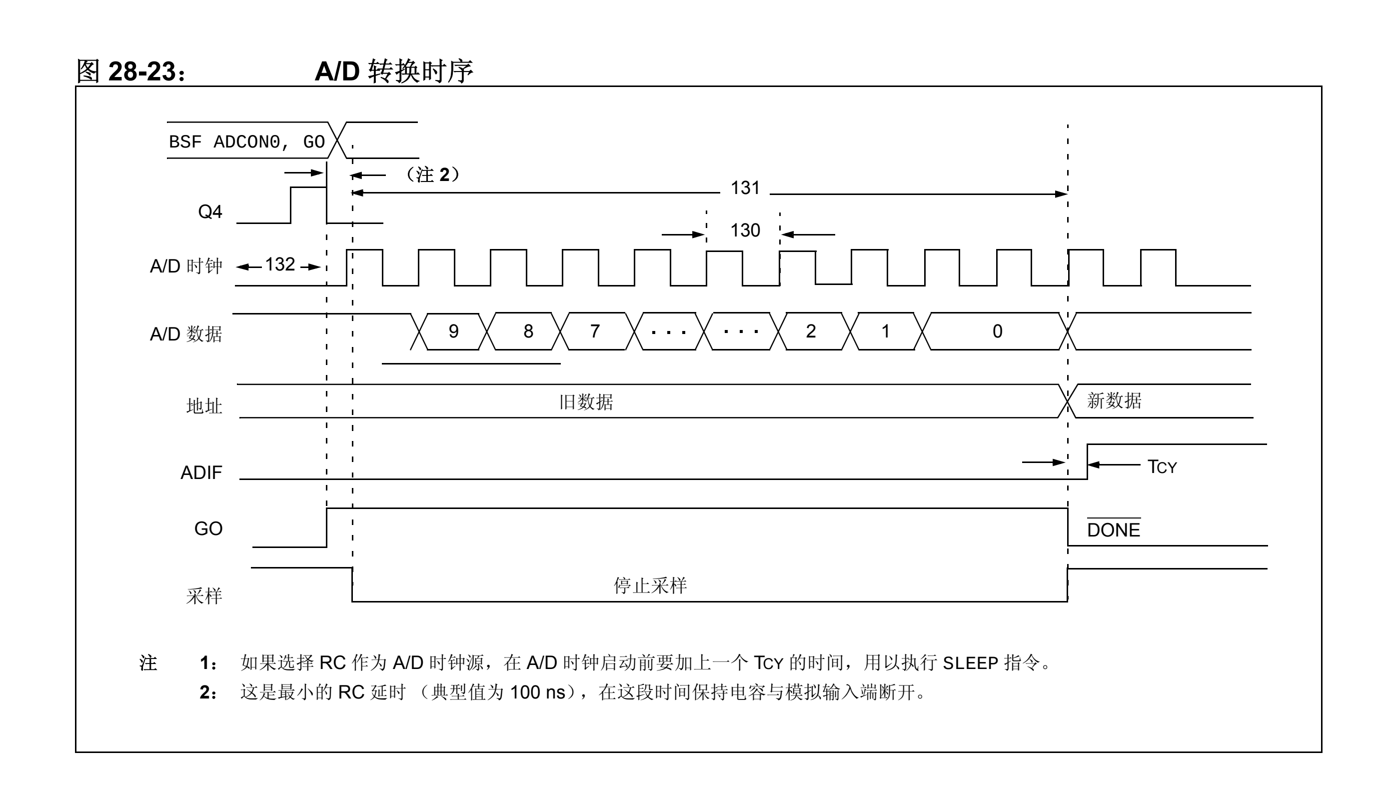
参数编号	符号	特性	最小值	最大值	单位	条件
	ToeF2adR	$\overline{\text{OESPP}}$ 下降沿到 CSSPP 上升沿的时间 （地址输出时间）	0	5	ns	
	ToeF2adV	$\overline{\text{OESPP}}$ 下降沿到地址输出有效的时间	0	5	ns	
	ToeR2adI	$\overline{\text{OESPP}}$ 上升沿到地址输出无效的时间	0	5	ns	
	ToeF2daR	$\overline{\text{OESPP}}$ 下降沿到 CSSPP 上升沿的时间 （数据输出时间）	0	5	ns	
	ToeF2daV	$\overline{\text{OESPP}}$ 下降沿到地址输出有效的时间	0	5	ns	
	ToeR2daI	$\overline{\text{OESPP}}$ 上升沿到数据输出无效的时间	0	5	ns	

表 28-28： A/D 转换器规范： PIC18F2455/2550/4455/4550 （工业级）
PIC18LF2455/2550/4455/4550（工业级）

参数编号	符号	特性	最小值	典型值	最大值	单位	条件
A01	NR	分辨率	—	—	10	位	ΔVREF ≥ 3.0V
A03	EIL	积分线性误差	—	—	<±1	LSb	ΔVREF ≥ 3.0V
A04	EDL	差分线性误差	—	—	<±1	LSb	ΔVREF ≥ 3.0V
A06	EOFF	偏移误差	—	—	<±1.5	LSb	ΔVREF ≥ 3.0V
A07	EGN	增益误差	—	—	<±1	LSb	ΔVREF ≥ 3.0V
A10	—	单调性	保证 (1)			—	VSS ≤ VAIN ≤ VREF
A20	ΔVREF	参考电压范围 （VREFH – VREFL）	1.8 3	— —	— —	V V	VDD < 3.0V VDD ≥ 3.0V
A21	VREFH	参考电压高电平	VSS	—	VREFH	V	
A22	VREFL	参考电压低电平	VSS-0.3V	—	VDD – 3.0V	V	
A25	VAIN	模拟输入电压	VREFL	—	VREFH	V	
A30	ZAIN	模拟电源推荐阻抗	—	—	2.5	kΩ	
A50	IREF	VREF 输入电流 (2)	— —	— —	5 150	μA μA	在采集 VAIN 期间。 在 A/D 转换期间。

注 1： A/D 转换结果不会因输入电压的增加而减小，并且不会丢失编码。
2： VREFH 电流来自作为 VREFH 源的 RA3/AN3/VREF+ 引脚或 VDD。
VREFL 电流来自作为 VREFL 源的 RA2/AN2/VREF-/CVREF 引脚或 VSS。

图 28-23： A/D 转换时序



注 1： 如果选择 RC 作为 A/D 时钟源，在 A/D 时钟启动前要加上一个 TCY 的时间，用以执行 SLEEP 指令。
2： 这是最小的 RC 延时 （典型值为 100 ns），在这段时间保持电容与模拟输入端断开。

表 28-29： A/D 转换要求

参数编号	符号	特性		最小值	最大值	单位	条件
130	TAD	A/D 时钟周期	PIC18**F**XXXX	0.7	25.0[1]	μs	基于 TOSC， VREF ≥ 3.0V
			PIC18**LF**XXXX	1.4	25.0[1]	μs	VDD = 2.0V； 基于 TOSC， VREF 满量程
			PIC18**F**XXXX	TBD	1	μs	A/D RC 模式
			PIC18**LF**XXXX	TBD	3	μs	VDD = 2.0V； A/D RC 模式
131	TCNV	转换时间 （不包括采集时间）（注 **2**）		11	12	TAD	
132	TACQ	采集时间 （注 **3**）		1.4 TBD	— —	μs μs	-40°C 至 +85°C 0°C 至 +85°C
135	TSWC	转换→采样的切换时间		—	（注 **4**）		
137	TDIS	电容放电时间		0.2	—	μs	

图注： TBD = 待定

注 **1**： A/D 时钟周期取决于器件频率和 TAD 时钟分频比。

2： 可在后续 TCY 周期内读 ADRES 寄存器。

3： 转换完成后当电压满幅变化时 （VDD 至 VSS，或 VSS 至 VDD），保持电容采集一个 “新”输入电压所需的时间。在输入通道上的电源阻抗 （*R*S）为 50Ω。

4： 在器件时钟的下一个周期上。

29.0 直流和交流特性图表

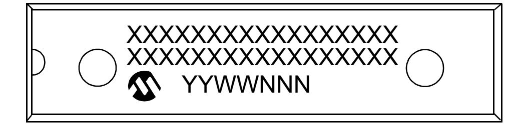
当前没有可用图表。

注：

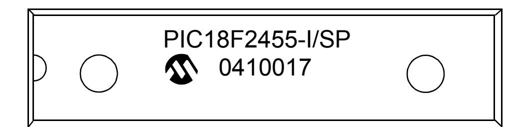
30.0 封装信息

30.1 封装标识信息

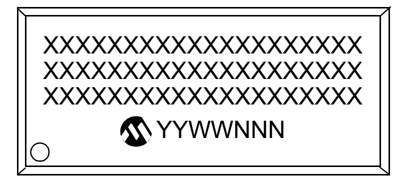
28 引脚 PDIP（窄型 DIP）



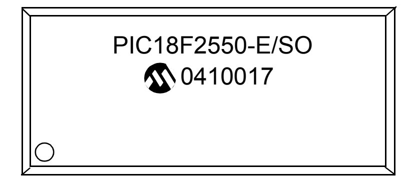
示例



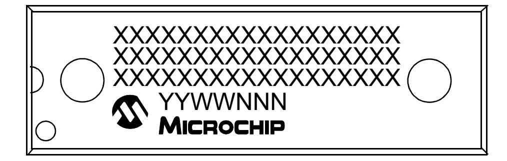
28 引脚 SOIC



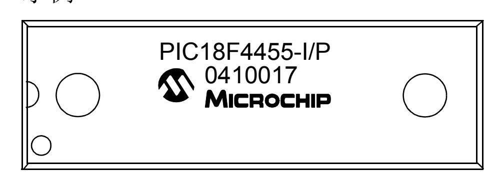
示例



40 引脚 PDIP



示例



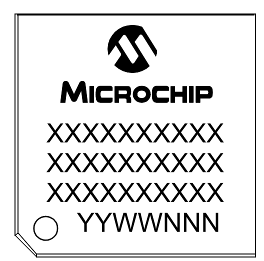
图注：		
	XX...X	客户特定信息 *
	Y	年份代码 （日历年的最后一位数字）
	YY	年份代码 （日历年的最后两位数字）
	WW	星期代码 （1 月 1 日的星期代码为 01）
	NNN	以字母数字排序的追踪代码

注： 若 Microchip 器件编号未在一行中完全标出，它将换行继续标出，因此会限制客户特定信息的可用字符数。

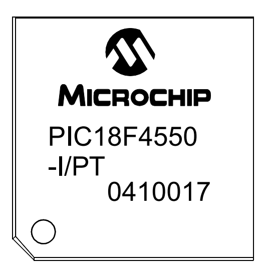
* 标准 PICmicro 器件标识包括 Microchip 器件编号、年份代码、星期代码和追踪代码。如需超过此范围的 PICmicro 器件标识，需支付一定的附加费用。请联系当地 Microchip 销售办事处，确认相关信息。对于 QTP 器件，在 QTP 价格中已包括任何特殊标记的加价。

封装标识信息 （续）

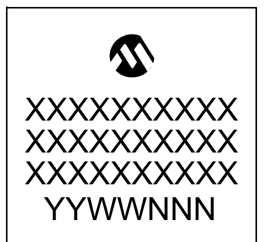
44 引脚 TQFP



示例



44 引脚 QFN



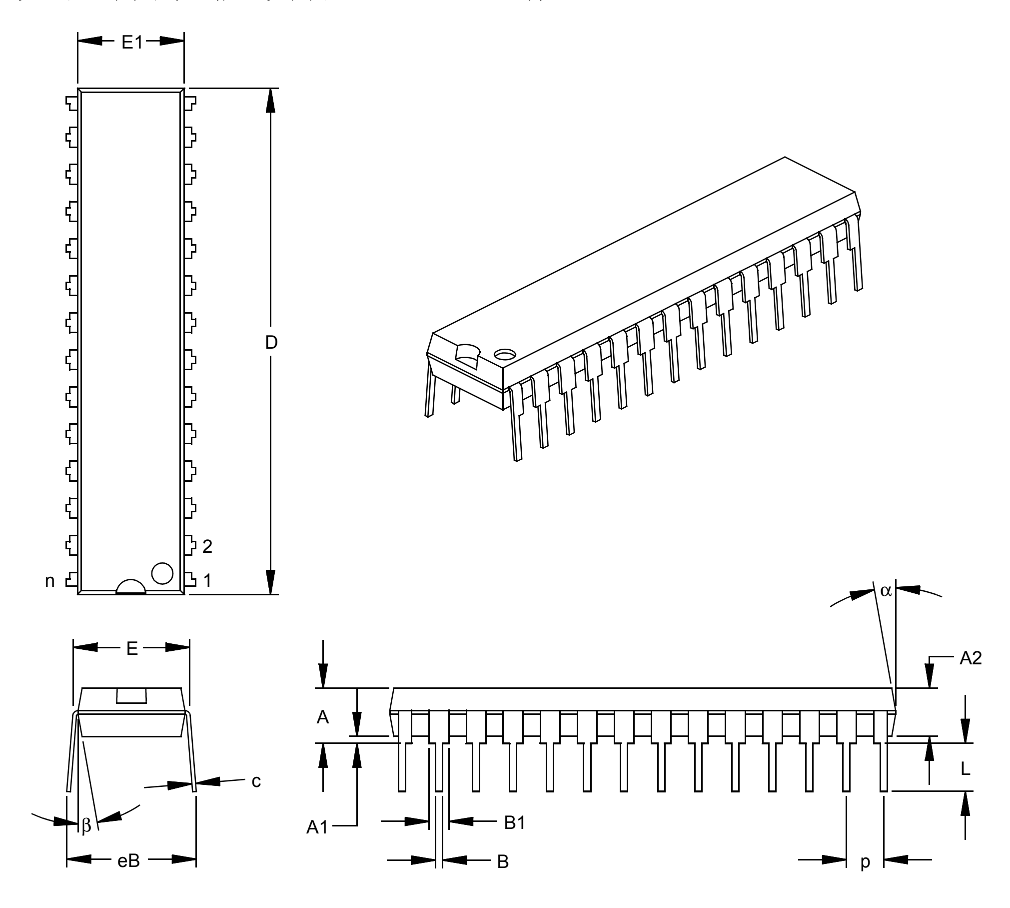
示例



30.2 封装详细信息

以下部分将介绍各种封装的技术细节。

28 引脚窄型塑封双列直插式封装 （SP）——主体 300 mil （PDIP）

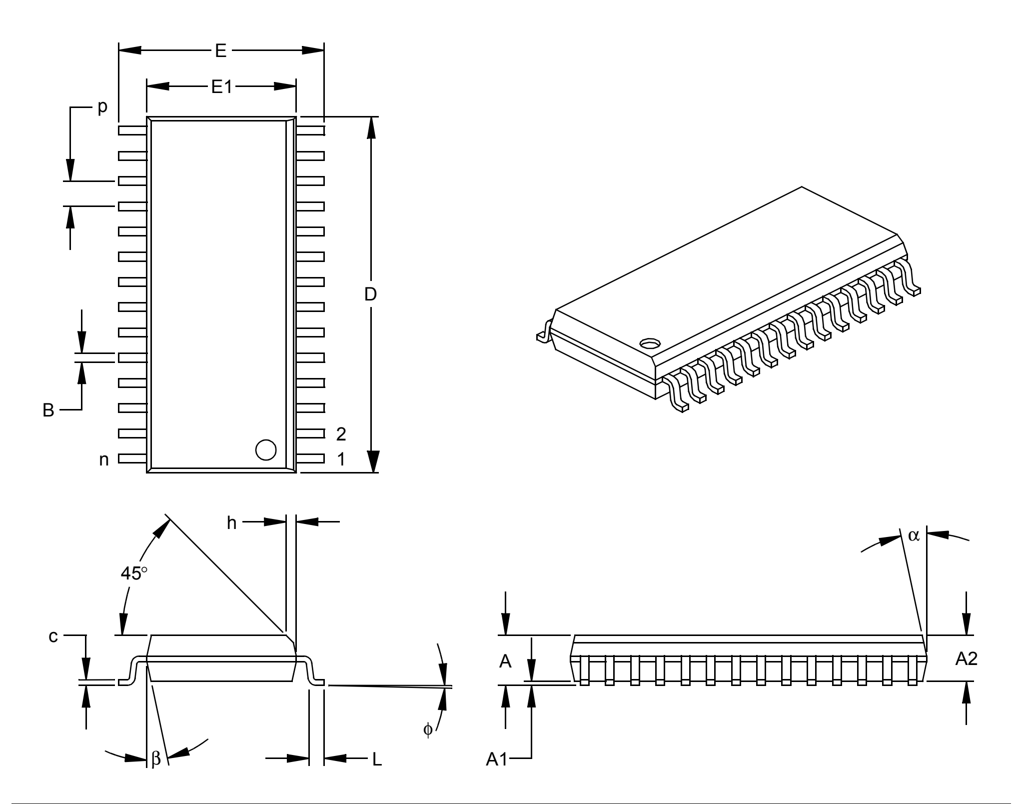


	单位	英寸 *			毫米		
	尺寸范围	最小	正常	最大	最小	正常	最大
引脚数	n		28			28	
引脚间距	p		.100			2.54	
顶端到固定面高度	A	.140	.150	.160	3.56	3.81	4.06
塑模封装厚度	A2	.125	.130	.135	3.18	3.30	3.43
塑模底面到固定面高度	A1	.015			0.38		
肩到肩宽度	E	.300	.310	.325	7.62	7.87	8.26
塑模封装宽度	E1	.275	.285	.295	6.99	7.24	7.49
总长度	D	1.345	1.365	1.385	34.16	34.67	35.18
引脚尖到固定面高度	L	.125	.130	.135	3.18	3.30	3.43
引脚厚度	c	.008	.012	.015	0.20	0.29	0.38
引脚上部宽度	B1	.040	.053	.065	1.02	1.33	1.65
引脚下部宽度	B	.016	.019	.022	0.41	0.48	0.56
总排列间距 §	eB	.320	.350	.430	8.13	8.89	10.92
塑模顶部锥度	α	5	10	15	5	10	15
塑模底部锥度	β	5	10	15	5	10	15

* 控制参数
§ 重要特性
注：
尺寸 D 和 E1 不包括塑模毛边或突起。塑模每侧的毛边或突起不得超过 0.010 英寸（0.254 毫米）。

等同于 JEDEC 号：MO-095
图号：C04-070

28 引脚塑封小外形封装 （SO）——主体 300 mil （SOIC）



单位		英寸 *			毫米		
尺寸范围		最小	正常	最大	最小	正常	最大
引脚数	n		28			28	
引脚间距	p		.050			1.27	
总高度	A	.093	.099	.104	2.36	2.50	2.64
塑模封装厚度	A2	.088	.091	.094	2.24	2.31	2.39
悬空间隙 §	A1	.004	.008	.012	0.10	0.20	0.30
总宽度	E	.394	.407	.420	10.01	10.34	10.67
塑模封装宽度	E1	.288	.295	.299	7.32	7.49	7.59
总长度	D	.695	.704	.712	17.65	17.87	18.08
斜面投影距离	h	.010	.020	.029	0.25	0.50	0.74
底脚长度	L	.016	.033	.050	0.41	0.84	1.27
底脚倾角	φ	0	4	8	0	4	8
引脚厚度	c	.009	.011	.013	0.23	0.28	0.33
引脚宽度	B	.014	.017	.020	0.36	0.42	0.51
塑模顶部锥度	α	0	12	15	0	12	15
塑模底部锥度	β	0	12	15	0	12	15

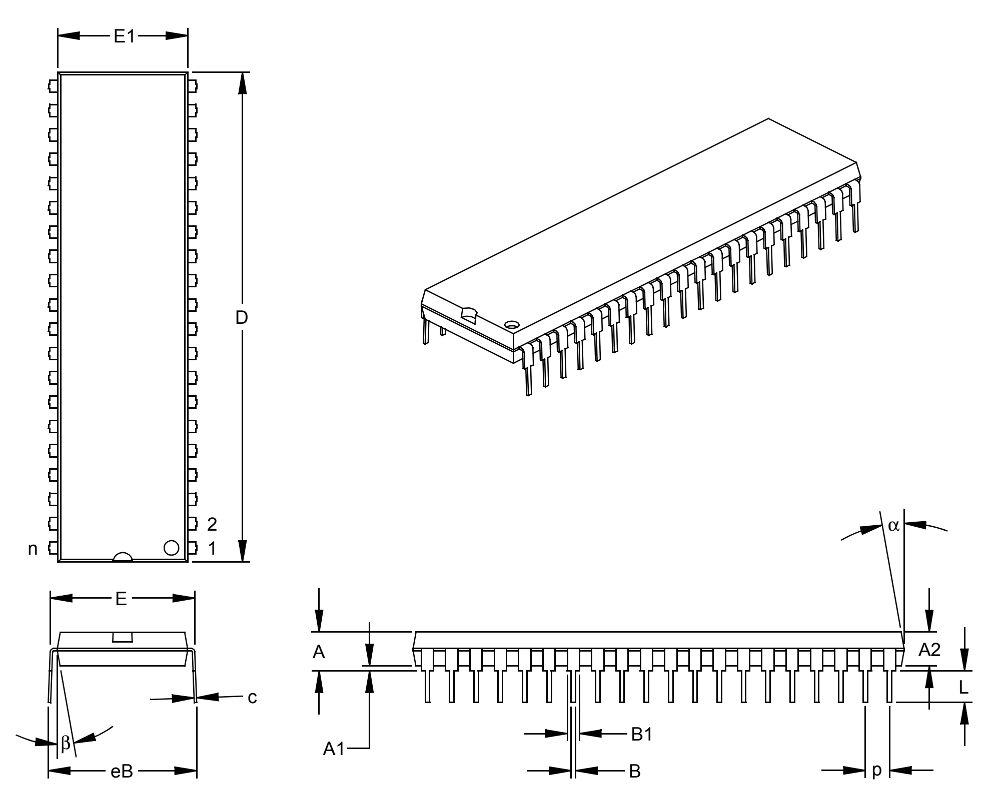
* 控制参数
§ 重要特性

注：
尺寸 D 和 E1 不包括塑模毛边或突起。塑模每侧的毛边或突起不得超过 0.010 英寸（0.254 毫米）。

等同于 JEDEC 号：MS-013
图号：C04-052

 初稿

40 引脚塑封双列直插式封装 （P）——主体 600 mil （PDIP）

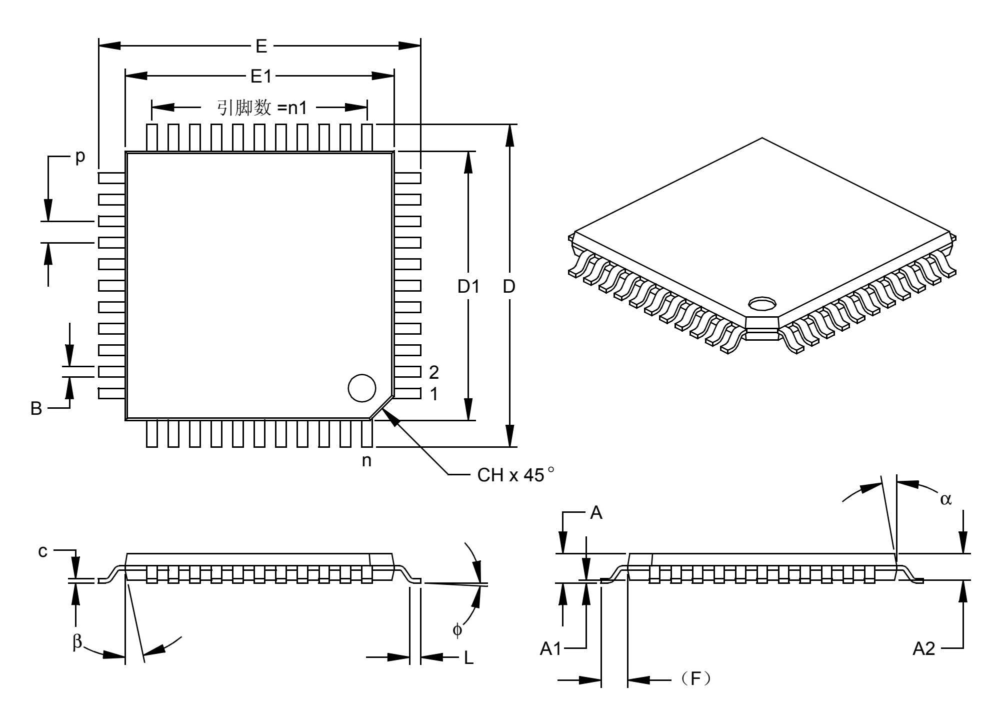


单位		英寸 *			毫米		
尺寸范围		最小	正常	最大	最小	正常	最大
引脚数	n		40			40	
引脚间距	p		.100			2.54	
顶端到固定面高度	A	.160	.175	.190	4.06	4.45	4.83
塑模封装厚度	A2	.140	.150	.160	3.56	3.81	4.06
塑模底面到固定面高度	A1	.015			0.38		
肩到肩宽度	E	.595	.600	.625	15.11	15.24	15.88
塑模封装宽度	E1	.530	.545	.560	13.46	13.84	14.22
总长度	D	2.045	2.058	2.065	51.94	52.26	52.45
引脚尖到固定面高度	L	.120	.130	.135	3.05	3.30	3.43
引脚厚度	c	.008	.012	.015	0.20	0.29	0.38
引脚上部宽度	B1	.030	.050	.070	0.76	1.27	1.78
引脚下部宽度	B	.014	.018	.022	0.36	0.46	0.56
总排列间距 §	eB	.620	.650	.680	15.75	16.51	17.27
塑模顶部锥度	α	5	10	15	5	10	15
塑模底部锥度	β	5	10	15	5	10	15

* 控制参数
§ 重要特性
注：
尺寸 D 和 E1 不包括塑模毛边或突起。塑模每侧的毛边或突起不得超过 0.010 英寸（0.254 毫米）。

等同于 JEDEC 号：MO-011
图号：C04-016

44 引脚塑封薄型正方扁平封装 （PT）——主体 10x10x1 mm，引脚形式 1.0/0.10 mm （TQFP）



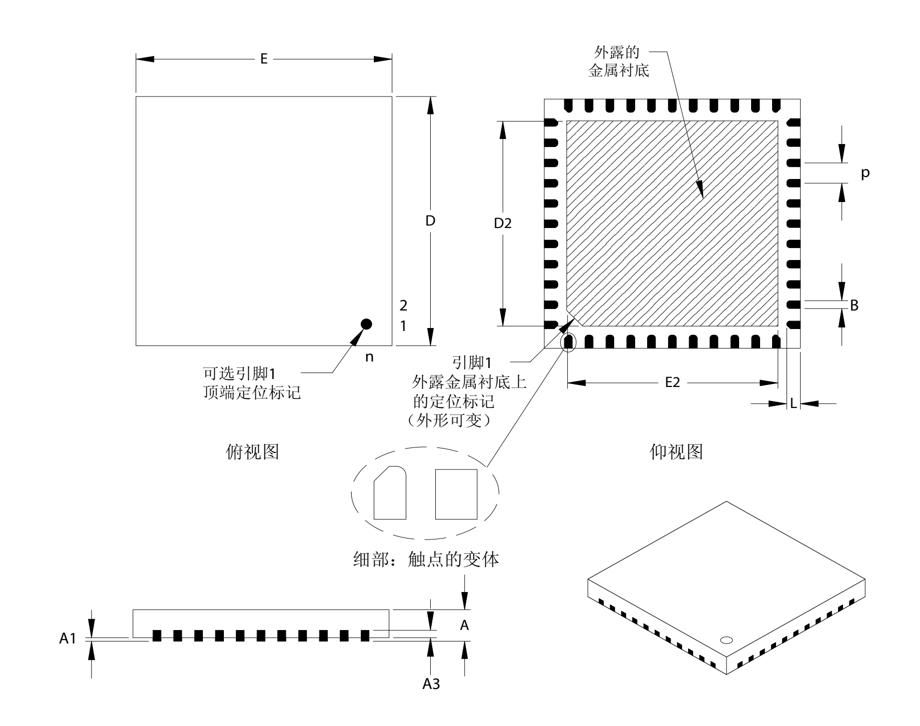
	单位	英寸			毫米 *		
	尺寸范围	最小	正常	最大	最小	正常	最大
引脚数	n		44			44	
引脚间距	p		.031			0.80	
每侧引脚数	n1		11			11	
总高度	A	.039	.043	.047	1.00	1.10	1.20
塑模封装厚度	A2	.037	.039	.041	0.95	1.00	1.05
悬空间隙 §	A1	.002	.004	.006	0.05	0.10	0.15
底脚长度	L	.018	.024	.030	0.45	0.60	0.75
底脚占位（参考）	（F）		.039		1.00		
底脚倾角	φ	0	3.5	7	0	3.5	7
总宽度	E	.463	.472	.482	11.75	12.00	12.25
总长度	D	.463	.472	.482	11.75	12.00	12.25
塑模封装宽度	E1	.390	.394	.398	9.90	10.00	10.10
塑模封装长度	D1	.390	.394	.398	9.90	10.00	10.10
引脚厚度	c	.004	.006	.008	0.09	0.15	0.20
引脚宽度	B	.012	.015	.017	0.30	0.38	0.44
引脚 1 处角斜面	CH	.025	.035	.045	0.64	0.89	1.14
塑模顶部锥度	α	5	10	15	5	10	15
塑模底部锥度	β	5	10	15	5	10	15

* 控制参数
§ 重要特性

注：
尺寸 D 和 E1 不包括塑模毛边或突起。塑模每侧的毛边或突起不得超过 0.010 英寸（0.254 毫米）。

等同于 JEDEC 号：MS-026
图号：C04-076

44 引脚塑封四方扁平无脚封装 （ML）——主体 8x8 mm （QFN）



单位		英寸			毫米*		
尺寸范围		最小	正常	最大	最小	正常	最大
触点数	n		44			44	
触点间距	p		.026 BSC [1]			0.65 BSC [1]	
总高度	A	.031	.035	.039	0.80	0.90	1.00
悬空间隙	A1	.000	.001	.002	0	0.02	0.05
基座厚度	A3		.010 REF [2]			0.25 REF [2]	
总宽度	E	.309	.315	.321	7.85	8.00	8.15
外露金属衬底宽度	E2	.236	.258	.260	5.99	6.55	6.60
总长度	D	.309	.315	.321	7.85	8.00	8.15
外露金属衬底长度	D2	.236	.258	.260	5.99	6.55	6.60
触点宽度	B	.008	.013	.013	0.20	0.33	0.35
触点长度	L	.014	.016	.019	0.35	0.40	0.48

*控制参数

注：

1. BSC：基本尺寸。理论上显示的是没有公差的精确值。
   参见ASME Y14.5M
2. REF：参考尺寸，通常无公差，仅供参考。
   参见ASME Y14.5M

外露金属衬底尺寸随管芯叶片大小而变化。

等同于JEDEC号：MO-220

图号C04-103，修订于05-05-05

注：

附录 A：版本历史

版本 A（2004 年 5 月）

PIC18F2455/2550/4455/4550 器件数据手册的最初版本。

版本 B（2004 年 10 月）

此版本对**第 28.0 节 "电气规范"**进行了更新，并对数据手册中的少量文本错误作了修订。

附录 B：器件差异

表 B-1 为本数据手册中所列器件间的差异。

表 B-1：器件差异

功能特性	PIC18F2455	PIC18F2550	PIC18F4455	PIC18F4550
程序存储器（字节）	24576	32768	24576	32768
程序存储器（指令）	12288	16384	12288	16384
中断源	19	19	20	20
I/O 端口	端口 A、B、C 或（E）	端口 A、B、C 或（E）	端口 A、B、C、D 和 E	端口 A、B、C、D 和 E
捕捉 / 比较 /PWM 模块	2	2	1	1
增强型捕捉 / 比较 /PWM 模块	0	0	1	1
并行通信（SPP）	无	无	有	有
10 位模数转换模块	10 路输入通道	10 路输入通道	13 路输入通道	13 路输入通道
封装	28 引脚 PDIP 28 引脚 SOIC	28 引脚 PDIP 28 引脚 SOIC	40 引脚 PDIP 44 引脚 TQFP 44 引脚 QFN	40 引脚 PDIP 44 引脚 TQFP 44 引脚 QFN

附录 C：转换注意事项

本附录讨论了器件从老版本升级到数据手册中所列版本时的注意事项。这些变化通常是由于采用的工艺技术不同而引起的。从 PIC16C74A 至 PIC16C74B 的升级就是一个例子。

不适用

附录 D：从基本型器件移植到增强型器件

本节讨论如何从基本型器件（即 PIC16C5X）移植到增强型 MCU 器件 （即 PIC18FXXX）。

下表列出了 PIC16C5X 单片机系列所做的修改：

当前没有数据

附录 E： 从中档器件移植到增强型器件

在 *AN716 “Migrating Designs from PIC16C74A/74B to PIC18C442”* 中详细讨论了中档 MCU 器件（即 PIC16CXXX）和增强型器件（即 PIC18FXXX）之间的差异。虽然所讨论的内容是针对特定器件的，但是适用于从中档器件移植至增强型器件的所有情况。

上述应用笔记的文献编号为 DS00716。

附录 F： 从高档器件移植到增强型器件

在 *AN726 “PIC17CXXX to PIC18CXXX Migration”* 中详细讨论了从高档 MCU 器件（即 PIC17CXXX）移植到增强型器件（即 PIC18FXXX）的步骤以及两者之间的差异。上述应用笔记的文献编号为 DS00726。

注：

索引

D

E

F

G

 初稿

K

L

M

 初稿

N

P

Q

R

U

W

X

Y

初稿

Z



注：

注：

MICROCHIP 网站

Microchip 网站（www.microchip.com）为客户提供在线支持。客户可通过该网站方便地获取文件和信息。只要使用常用的因特网浏览器即可访问。网站提供以下信息：

- **产品支持**——数据手册和勘误表、应用笔记和样本程序、设计资源、用户指南以及硬件支持文档、最新的软件版本以及存档软件
- **一般技术支持**——常见问题（FAQ）、技术支持请求、在线讨论组以及 Microchip 顾问计划成员名单
- **Microchip 业务**——产品选型和订购指南、最新 Microchip 新闻稿、研讨会和活动安排表、Microchip 销售办事处、代理商以及工厂代表列表

变更通知客户服务

Microchip 的变更通知客户服务有助于客户了解 Microchip 产品的最新信息。注册客户可在他们感兴趣的某个产品系列或开发工具发生变更、更新、发布新版本或勘误表时，收到电子邮件通知。

欲注册，请登录 Microchip 网站 www.microchip.com，点击"变更通知客户（Customer Change Notification）"服务后按照注册说明完成注册。

客户支持

Microchip 产品的用户可通过以下渠道获得帮助：

- 代理商或代表
- 当地销售办事处
- 应用工程师（FAE）
- 技术支持

客户应联系其代理商、代表或应用工程师（FAE）寻求支持。当地销售办事处也可为客户提供帮助。本文档后附有销售办事处的联系方式。

也可通过 http：//support.microchip.com 获得网上技术支持。

读者反馈表

我们努力为您提供最佳文档，以确保您能够成功使用 Microchip 产品。如果您对文档的组织、条理性、主题及其他有助于提高文档质量的方面有任何意见或建议，请填写本反馈表并传真给我公司 TRC 经理，传真号码为 86-21-5407-5066。

请填写以下信息，并从下面各方面提出您对本文档的意见。

致： TRC 经理　　　　　　　　　　　　　　总页数 ________

关于： 读者反馈

发自： 姓名 ________________________________

公司 ________________________________

地址 ________________________________

国家 / 省份 / 城市 / 邮编 ________________________

电话：（______） ________________　　　　传真：（______） ________________

应用（选填）：

您希望收到回复吗？是____ 否____

器件： PIC18F2455/2550/4455/4550　文献编号： DS39632B_CN

问题

1. 本文档中哪些部分最有特色？

____

____

2. 本文档是否满足了您的软硬件开发要求？如何满足的？

____

____

3. 您认为本文档的组织结构便于理解吗？如果不便于理解，那么问题何在？

____

____

4. 您认为本文档应该添加哪些内容以改善其结构和主题？

____

____

5. 您认为本文档中可以删减哪些内容，而又不会影响整体使用效果？

____

____

6. 本文档中是否存在错误或误导信息？如果存在，请指出是什么信息及其具体页数。

____

____

7. 您认为本文档还有哪些方面有待改进？

____

____

PIC18F2455/2550/4455/4550 产品标识体系

欲订货或获取价格、交货等信息，请与我公司生产厂或销售办事处联系。

器件编号	X	/XX	XXX
器件	温度范围	封装	模式

器件	PIC18F2455/2550[1] 和 PIC18F4455/4550[1]， PIC18F2455/2550T[2] 和 PIC18F4455/4550T[2]； VDD 范围为 4.2V 至 5.5V PIC18LF2455/2550[1] 和 PIC18LF4455/4550[1]， PIC18LF2455/2550T[2] 和 PIC18LF4455/4550T[2]； VDD 范围为 2.0V 至 5.5V
温度范围	I = -40°C 至 +85°C （工业级） E = -40°C 至 +125°C （扩展级）
封装	PT = TQFP （薄型正方扁平封装） SO = SOIC SP = 窄型塑封 DIP P = PDIP ML = QFN
模式	QTP、SQTP、编码或特殊要求 （空白为其他情况）

示例：

a) PIC18LF4550-I/P 301 = 工业级温度， PDIP 封装，扩展级 VDD 范围和 QTP 模式 #301。

b) PIC18LF2455-I/SO = 工业级温度， SOIC 封装和扩展级 VDD 范围。

c) PIC18F4455-I/P = 工业级温度， PDIP 封装和普通 VDD 范围。

注 1： F = 标准电压范围
LF = 宽电压范围

2： T = 卷带式封装（仅TQFP封装）。



全球销售及服务网点

美洲

公司总部 **Corporate Office**
2355 West Chandler Blvd.
Chandler, AZ 85224-6199
Tel: 1-480-792-7200
Fax: 1-480-792-7277
技术支持：
http://support.microchip.com
网址：www.microchip.com

亚太总部 **Asia Pacific Office**
Suites 3707-14, 37th Floor
Tower 6, The Gateway
Habour City, Kowloon
Hong Kong
Tel: 852-2401-1200
Fax: 852-2401-3431

亚特兰大 **Atlanta**
Alpharetta, GA
Tel: 1-770-640-0034
Fax: 1-770-640-0307

波士顿 **Boston**
Westborough, MA
Tel: 1-774-760-0087
Fax: 1-774-760-0088

芝加哥 **Chicago**
Itasca, IL
Tel: 1-630-285-0071
Fax: 1-630-285-0075

达拉斯 **Dallas**
Addison, TX
Tel: 1-972-818-7423
Fax: 1-972-818-2924

底特律 **Detroit**
Farmington Hills, MI
Tel: 1-248-538-2250
Fax: 1-248-538-2260

科科莫 **Kokomo**
Kokomo, IN
Tel: 1-765-864-8360
Fax: 1-765-864-8387

洛杉矶 **Los Angeles**
Mission Viejo, CA
Tel: 1-949-462-9523
Fax: 1-949-462-9608

圣何塞 **San Jose**
Mountain View, CA
Tel: 1-650-215-1444
Fax: 1-650-961-0286

加拿大多伦多 **Toronto**
Mississauga, Ontario,
Canada
Tel: 1-905-673-0699
Fax: 1-905-673-6509

亚太地区

中国 – 北京
Tel: 86-10-8528-2100
Fax: 86-10-8528-2104

中国 – 成都
Tel: 86-28-8676-6200
Fax: 86-28-8676-6599

中国 – 福州
Tel: 86-591-8750-3506
Fax: 86-591-8750-3521

中国 – 香港特别行政区
Tel: 852-2401-1200
Fax: 852-2401-3431

中国 – 青岛
Tel: 86-532-8502-7355
Fax: 86-532-8502-7205

中国 – 上海
Tel: 86-21-5407-5533
Fax: 86-21-5407-5066

中国 – 沈阳
Tel: 86-24-2334-2829
Fax: 86-24-2334-2393

中国 – 深圳
Tel: 86-755-8203-2660
Fax: 86-755-8203-1760

中国 – 顺德
Tel: 86-757-2839-5507
Fax: 86-757-2839-5571

中国 – 武汉
Tel: 86-27-5980-5300
Fax: 86-27-5980-5118

中国 – 西安
Tel: 86-29-8833-7250
Fax: 86-29-8833-7256

台湾地区 – 高雄
Tel: 886-7-536-4818
Fax: 886-7-536-4803

台湾地区 – 台北
Tel: 886-2-2500-6610
Fax: 886-2-2508-0102

台湾地区 – 新竹
Tel: 886-3-572-9526
Fax: 886-3-572-6459

亚太地区

澳大利亚 **Australia - Sydney**
Tel: 61-2-9868-6733
Fax: 61-2-9868-6755

印度 **India - Bangalore**
Tel: 91-80-4182-8400
Fax: 91-80-4182-8422

印度 **India - New Delhi**
Tel: 91-11-5160-8631
Fax: 91-11-5160-8632

印度 **India - Pune**
Tel: 91-20-2566-1512
Fax: 91-20-2566-1513

日本 **Japan - Yokohama**
Tel: 81-45-471- 6166
Fax: 81-45-471-6122

韩国 **Korea - Gumi**
Tel: 82-54-473-4301
Fax: 82-54-473-4302

韩国 **Korea - Seoul**
Tel: 82-2-554-7200
Fax: 82-2-558-5932 或
82-2-558-5934

马来西亚 **Malaysia - Penang**
Tel: 60-4-646-8870
Fax: 60-4-646-5086

菲律宾 **Philippines - Manila**
Tel: 63-2-634-9065
Fax: 63-2-634-9069

新加坡 **Singapore**
Tel: 65-6334-8870
Fax: 65-6334-8850

泰国 **Thailand - Bangkok**
Tel: 66-2-694-1351
Fax: 66-2-694-1350

欧洲

奥地利 **Austria - Wels**
Tel: 43-7242-2244-3910
Fax: 43-7242-2244-393

丹麦 **Denmark-Copenhagen**
Tel: 45-4450-2828
Fax: 45-4485-2829

法国 **France - Paris**
Tel: 33-1-69-53-63-20
Fax: 33-1-69-30-90-79

德国 **Germany - Munich**
Tel: 49-89-627-144-0
Fax: 49-89-627-144-44

意大利 **Italy - Milan**
Tel: 39-0331-742611
Fax: 39-0331-466781

荷兰 **Netherlands - Drunen**
Tel: 31-416-690399
Fax: 31-416-690340

西班牙 **Spain - Madrid**
Tel: 34-91-708-08-90
Fax: 34-91-708-08-91

英国 **UK - Wokingham**
Tel: 44-118-921-5869
Fax: 44-118-921-5820

06/08/06